

**RECHENZENTRUM TH MÜNCHEN**  
**ARBEITSGRUPPE BETRIEBSSYSTEME**

**INTERNSCHRIFT Nr. 44**

**THEMA**            Zur Zeitmessung im BSM

**VERFASSTER**    Bader

**DATUM**        8.7.1970

**FORM DER ABFASSUNG**

Entwurf

☒ Ausarbeitung

Endform

**SACHLICHE VERBINDLICHKEIT**

Allgemeine Information  
Diskussionsgrundlage

☒ Erarbeiteter Vorschlag

Verbindliche Mitteilung

☒ Veraltet

**ÄNDERUNGSZUSTAND**

**BEZUG AUF INTERNSCHRIFTEN**

**ANDERWEITIGE LITERATUR**

Information über Rechnerhalte:

"Doppel-TR4-System auf TR440, Bedienungshandbuch  
(AEG-Telefunken Datenverarbeitung, Beschreibung  
Nr. N3/GR64)"

---

Arbeitsunterlage, nicht zur Publikation bestimmt. Weitergabe  
an Dritte nur im Einvernehmen mit der Arbeitsgruppe

Zur Zeitmessung im BSM

## Inhalt:

1. Einleitung	
1.1 Aufgabenabgrenzung	1
1.2 Hardwaregegebenheiten	2
1.3 Voraussetzungen	3
1.3.1 Weckeralarme	3
1.3.2 Rechnerhalte	3
1.4 Vorbemerkung	3
2. Systemzeit	3
2.1 Definition der Systemzeit	3
2.2 Berechnung der Systemzeit	4
2.3 Languhrberechnung	5
2.4 Fortschreibung der Languhraddenden	6
2.5 Realisierung eines Zeitabgleichs	7
2.5.1 Zeitabgleichstermin	7
2.5.2 Zeitabgleichssperre	7
2.5.3 Vorbereitung des Zeittests	7
2.5.4 Zeittest	8
2.5.5 Nachholung einseitig versäumter Uhrumläufe	9
2.5.6 Zeitabgleich	10
2.5.7 Beendigung der Zeitpflegearbeiten	10
2.6 Die drei "Uhrparameter"	11
2.6.1 uhrparameter1 = 15	11
2.6.2 uhrparameter2 = 3	11
2.6.3 uhrparameter3 = 84000	11
2.7 Brauchbarkeit der simulierten Systemuhr	12

2.8 Auswirkung von Rechnerhalten auf die Zeitverfolgung	14
2.8.1 Einseitiger Rechnerhalt	15
2.8.2 Zweiseitiger Rechnerhalt	15
2.8.3 Brauchbarkeit der simulierten Systemuhr auch bei Rechnerhalten	15
2.9 Programmbeschreibungen	16
2.9.1 Zeitberechnungsprogramm	16
2.9.2 Zeitpflegeprogramm1 (Teil der Weckeralarmbe- handlung)	19
2.9.3 Zeitpflegeprogramm2 (Teil der Rechnerkern- alarmbehandlung)	25
3. Verbuchung von Rechnerkernzeiten	26
3.1 Variable zur Messung des Verbrauchs an Rechner- kernzeit	26
3.2 Ablaufplan und TAS-Code zur Verbuchung der Rechner- kernzeit	27
Anhang 1. Zur Signalisierung zwischen Rechnerkernen via Kernspeicher	28



## 1. Einleitung

### 1.1 Aufgabenabgrenzung

In dieser Schrift werden die folgenden beiden Aufgaben, die zur Schicht 1 des BSM gehören, behandelt:

- a. Definition einer für beide Rechnerkerne einheitlichen "Systemzeit". Dies ist eine Größe, die in je 10  $\mu$ s um 1 Einheit zunimmt und nie überläuft. (Die Zeiteinheit von 10  $\mu$ s ist die gleiche wie bei den Hardwareuhren.)

Verfolgung der Systemzeit und damit auch der absoluten Zeit durch das BSM.

Bereitstellung eines Dienstes für eine schnelle Berechnung der Systemzeit.

- b. Verbuchung der von den Bearbeitern verbrauchten Rechnerkernzeit. (Die in Internschrift 39 darüber gemachten Ausführungen sind damit überholt.)

Auf andere zur Zeitmessung gehörige Aufgaben wird hier noch nicht eingegangen. Eine solche Aufgabe ist z.B. die Berechnung der in der Außenwelt gültigen Zeit (Angabe von Wochentag und Datum sowie der MEZ in Stunden, Minuten und evtl. auch Sekunden). Als Berechnungsgrundlage dafür reicht die Systemzeit allein nicht aus, vielmehr ist noch - nach dem Einschalten der Rechenanlage oder nach einem langen zweiseitigen Rechnerhalt - eine einmalige Eingabe des Operateurs notwendig, mit deren Hilfe die Korrespondenz zwischen Außenweltzeit und Systemzeit hergestellt wird. Diese Aufgabe muß vielleicht in einer höheren Schicht als 1 gelöst werden, weil dort die Koordinierung der Ein- und Ausgabe auf der Operateurkonsole erfolgt.



### 1.2 Hardwaregegebenheiten

Jeder Rechnerkern  $r$  ( $r=0, 1$ ) hat ein schwingquarzgesteuertes 24-stelliges Uhrregister (kurz: Uhr), das hier mit  $BG_r$  bezeichnet wird. Der Inhalt des Uhrregisters gilt als ganze Zahl;  $0 \leq \langle BG_r \rangle \leq 2^{24}-1$ . Der Schwingquarz veranlaßt, daß der Inhalt des zugehörigen Uhrregisters nach jeweils einer Zeiteinheit um 1 erhöht wird (modulo  $2^{24}$ ). Dabei gilt

$$1 \text{ Zeiteinheit} = (1 + \epsilon) \cdot 10 \text{ } \mu\text{s, mit } |\epsilon| \leq 10^{-5},$$

wobei  $\epsilon$  i.allg. für jede Zeiteinheit anders ist. Auch bei Rechnerhalt (der ja ein Laufen des Rechnerkerns in einer Mikroprogrammschleife ist und nicht mit "Rechner aus" verwechselt werden darf), geht die Hochzählung im Uhrregister weiter.

Es gibt nur eine einzige Möglichkeit, auf  $\langle BG_r \rangle$  per Programm zuzugreifen, nämlich durch Ausführung eines VMØ-Befehls (mit speziellem Adreßteil) durch Rechnerkern  $r$ ; dabei wird  $\langle BG_r \rangle$  in dessen Register B übertragen.

Die beiden Uhren laufen asynchron. Die Ganggeschwindigkeit einer Uhr weicht vom Sollwert (100 000 Zeiteinheiten pro sec) nie mehr als 1 Zeiteinheit pro sec ab. Damit ergibt sich eine Maximalabweichung von  $24 \cdot 60 \cdot 60$  Zeiteinheiten (entsprechend 0,86 Sekunden) pro Tag; die Ganggenauigkeit ist also sehr hoch.

Auf dem Rechnerkern  $r$  findet ein Uhrüberlauf, d.h. ein Wechsel von  $\langle BG_r \rangle = 2^{24}-1$  nach  $\langle BG_r \rangle = 0$ , nach jeweils  $2^{24}$  Zeiteinheiten statt. Die Größe von  $2^{24}$  Zeiteinheiten (ca. 2,8 min) möge im folgenden ein Uhrumlauf genannt werden.

Die Dauer eines Ablaufs, der in jedem Fall weniger Zeit braucht als ein Uhrumlauf, kann berechnet werden, wenn man den Uhrregisterinhalt am Anfang und Ende kennt (allerdings vom selben Rechnerkern her). Dieser Sachverhalt wird bei der Verbuchung der Rechnerkernzeiten (s. 3 ) ausgenutzt.

Kommen jedoch längerdauernde Abläufe ins Spiel, oder solche, bei denen der darunterliegende Rechnerkern wechseln kann, so reichen die Uhrregisterinhalte allein zur Zeitfeststellung nicht mehr aus. Es muß dann durch Programmatur eine "relative Maschinenzeit" wie z.B. die in Abschnitt 2 beschriebene Systemzeit zur Verfügung gestellt werden.

### 1.3 Voraussetzungen

#### 1.3.1 Weckeralarme

Während des BSM-Betriebs darf der Schalter "Wecker aus" des Bedienpultes nicht eingelegt werden und das Weckerregister BW darf höchstens innerhalb von Weckeralarmbehandlungen neu eingestellt werden.

Damit ist gewährleistet, daß spätestens nach jeweils ca. 0,66 sec ein Weckeralarm anzustehen kommt, der als auslösendes Moment für periodisch zu wiederholende Arbeiten benutzt werden kann.

#### 1.3.2 Rechnerhalte

Rechnerhalte sollen nach Möglichkeit im BSM-Betrieb nicht auftreten, sind hier jedoch, in den gängigen Fällen, mit eingeplant. Vgl. 2.8.

### 1.4 Vorbemerkung

Alle in dieser Schrift vorkommenden Variable und Konstanten sind ganzzahlig; soweit sie Zeiten anzeigen, bedeutet eine Einheit stets eine Zeiteinheit ( $\approx 10 \mu\text{s}$ ).

## 2. Systemzeit

### 2.1 Definition der Systemzeit

Die folgende Definition gilt für einen gewissen Zeitraum, im dem beide Rechnerkerne eingeschaltet sind. Dieser Zeitraum beginnt nach der Systemzeitfundierung, die unmittelbar nach dem Einschalten

der Rechenanlage vorgenommen wird, und endet mit dem nächsten Ausschalten der Rechenanlage.

$$\text{Systemzeit} = \text{zeitbasis} + \left[ \frac{\text{languhr}_0 + \text{languhr}_1}{2} \right]$$

Dabei ist zeitbasis ( $0 \leq \text{zeitbasis} \leq 2^{46}$ ) eine Konstante, die bei jeder neuen Systemzeitfundierung so festgelegt werden kann, daß sie größer ist als das Ergebnis aller früheren Systemzeitberechnungen. (Dadurch kann erreicht werden, daß die Systemzeit über mehr als 20 Jahre hinweg ein monoton-nichtfallende Funktion der (physikalischen) Zeit ist.) ( $2^{46}$  Zeiteinheiten sind mehr als 22 Jahre.)

languhr<sub>r</sub> ( $r=0, 1$ ) ist der Inhalt eines gedachten Registers des Rechnerkerns  $r$ , das aus BG<sub>r</sub> durch linksseitige Verlängerung um 22 Binärstellen entsteht (Languhr). languhr<sub>r</sub> möge in den rechten 24 Bits stets mit (BG<sub>r</sub>) übereinstimmen, und in den linken 22 Bits von languhr<sub>r</sub> möge bei jedem Uhrüberlauf von Rechnerkern  $r$  eine 1 addiert werden. Als anfänglicher Inhalt der linken 22 Bits von languhr<sub>r</sub> wird bei der Systemzeitfundierung für  $r=0,1$  jeweils eine Zahl  $\geq 0$ ,  $\ll 2^{22}$  vereinbart; damit gilt auch im ganzen betrachteten Zeitraum

$$0 \leq \text{languhr}_r \ll 2^{46}.$$

Die Systemzeit schreitet nach obiger Definition mit derselben Geschwindigkeit fort, mit der die beiden Hardwareuhren im Mittel laufen. Damit ergibt sich auch für die Systemzeit eine max. Ungenauigkeit von 0,86 sec pro Tag.

## 2.2 Berechnung der Systemzeit

Als Näherungswert für die Systemzeit wird berechnet

$$s_r'(t) := \text{languhr}_r'(t) + \text{languhraddend}_r.$$

Dabei ist  $r=0,1$  die Nummer des Rechnerkerns, auf dem die Systemzeitberechnung stattfindet (Rechnerkernwechsel während der Berechnung wird durch Alarmsperre 1 verhindert), languhr<sub>r</sub>'(t) ist das Ergebnis



einer frisch vorgenommenen Languhrberechnung (s. 2.2), deren zugehörige Uhrabfrage im (absoluten) Zeitpunkt  $t$  erfolgt, und der Languhraddend ist eine Näherung für die Abweichung zwischen Systemzeit und Languhr.

(Das Apostroph bei  $\text{languhr}_r'(t)$  ist für den Fall gesetzt, daß wegen langer zweiseitiger Rechnerhalte Uhrumläufe verpaßt wurden und der berechnete Languhrwert  $\text{languhr}_r(t)$  um Vielfache von  $2^{24}$  kleiner ist als der in 2.1 definierte Wert  $\text{languhr}_r(t)$ .)

Genauigkeit der berechneten Systemzeit (Unterlagen dazu sind beim Verfasser):

$$\left| s_r'(t) + k \cdot 2^{24} - \text{systemzeit}(t) \right| \leq 2 + \left\lceil \frac{\Delta t + \frac{1}{2} \text{ sec}}{\text{sec}} \right\rceil$$

Dabei ist  $\Delta t$  die seit dem letzten Zeitabgleich verstrichene Zeit und  $k=0,1,2,\dots$  ist die Anzahl der (wegen langer Rechnerhalte) verpaßten Uhrumläufe.

### 2.3 Languhrberechnung

Die Languhrberechnungen auf Rechnerkern  $r$  basieren darauf, daß das Ergebnis einer jeden Languhrberechnung in der Variablen  $\text{languhrprobe}_r$  festgehalten wird.

Bei einer Languhrberechnung wird zunächst im Zeitpunkt  $t$  der Inhalt der Uhr  $\text{BG}_r$  abgelesen und anschließend linksseitig mit 22 Bits derart ergänzt, daß für das Ergebnis  $\text{languhr}_r'(t)$  gilt

$$\text{languhrprobe}_r \leq \text{languhr}_r'(t) \leq \text{languhrprobe}_r + 2^{24} - 1.$$

Danach wird der neu berechnete Languhrwert als Grundlage für die nächste Languhrberechnung durch

$$\text{languhrprobe}_r := \text{languhr}_r'(t)$$

abgelegt.

Bei diesem Vorgehen ist sichergestellt, daß  $\text{languhrprobe}_r$  eine monoton-nichtfallende Funktion der Zeit ist. Wenn auf dem Rechnerkern  $r$  zwei aufeinanderfolgende Languhrberechnungen nie weiter als  $2^{24}-1$  Zeiteinheiten auseinanderliegen, gilt ferner für jede Berechnung  $\text{languhr}_r'(t) = \text{languhr}_r(t)$ .

#### 2.4 Fortschreibung der Languhraddenden

Die Größen  $\text{languhraddend}_0$  und  $\text{languhraddend}_1$  werden nur bei Zeitabgleichen verändert; die Änderungen sind stets geringfügig. Sofern Rechnerhalte dem nicht entgegenstehen, wird alle 0,84 bis 1,50 Sekunden (vgl. 2.6.3) zwischen beiden Rechnerkernen ein Zeitabgleich vorgenommen. Dazu führen Rechnerkern 0 und 1 nahezu gleichzeitig ( $|t_0 - t_1| < 9,9999 \mu\text{s}$ ) eine Languhrberechnung durch, mit den Ergebnissen  $\text{languhr}_0'(t_0)$  und  $\text{languhr}_1'(t_1)$ , woraus dann, ggf. nach Berücksichtigung der von einem Rechnerkern zusätzlich verpaßten Uhrumläufe, als Systemzeit für den "Abgleichszeitpunkt"  $t_A = \max(t_0, t_1)$  berechnet wird:

$$s' := \text{zeitbasis} + \left\lfloor \frac{\text{languhr}_0'(t_0) + \text{languhr}_1'(t_1)}{2} \right\rfloor$$

Es gilt dann:

$$\text{systemzeit}(t_A) = s' + k \cdot 2^{24} + \varepsilon,$$

$$\text{languhr}_0(t_0) = \text{languhr}_0'(t_0) + k \cdot 2^{24},$$

$$\text{languhr}_1(t_1) = \text{languhr}_1'(t_1) + k \cdot 2^{24}.$$

Dabei ist  $\varepsilon = 0$  oder  $\varepsilon = 1$  und  $k = 0, 1, 2, \dots$  ist die Anzahl der (wegen langer Rechnerhalts, s. 2.8.2) verpaßten Uhrumläufe. Die Korrektur der Languhraddenden wird dann für  $r = 0, 1$  durch

$\text{languhraddend}_r := s' - \text{languhr}_r'(t_r) = \text{systemzeit}(t_A) - \varepsilon - \text{languhr}_r(t_r)$  vorgenommen.

Der genaue Ablauf eines Zeitabgleichs wird im folgenden beschrieben.

## 2.5 Realisierung eines Zeitabgleichs

### 2.5.1 Zeitabgleichstermin

Bei jeder Weckeralarmbehandlung wird das Zeitpflegeprogramm 1 durchlaufen. Dabei wird zunächst eine Zeitberechnung vorgenommen. Deren Ergebnis  $s$  wird verglichen mit der Größe  $zatermin$ , die anzeigt, wann der nächste Zeitabgleich fällig ist. Nur für  $s \geq zatermin$  fallen dann noch weitere Arbeiten im Zeitpflegeprogramm 1 an. Es wird dann versucht, in eine Zeitabgleichs-Sperrphase einzutreten.

### 2.5.2 Zeitabgleichssperre

Die Zeitabgleichssperre verhindert Kollisionen und Verklemmungen, die entstehen könnten, wenn beide Rechnerkerne gleichzeitig einen Zeitabgleich vornehmen wollten und ist analog zur AKL-Sperre (s. Internschrift 39) organisiert: jeweils höchstens ein Rechnerkern kann in einer Zeitabgleichssperrphase sein.

Gelingt dem Rechnerkern  $r$  der Eintritt in eine Zeitabgleichssperrphase, so gilt er solange als Zeitmeister, bis er die Sperre wieder aufhebt. Mißglückt der Eintrittsversuch, so ist der andere Rechnerkern  $r'$  bereits Zeitmeister und  $r$  verläßt das Zeitpflegeprogramm 1 wieder; auf den bald zu erwartenden Rechnerkernalarm hin kann  $r$  dann die Rolle des "Zeitgehilfen" beim Zeitabgleich übernehmen.

### 2.5.3 Vorbereitung des Zeittests

Ist Rechnerkern  $r$  Zeitmeister geworden, so schickt er an Rechnerkern  $r'$  einen Rechnerkernalarm (mit der Zusatzinformation  $zeittestbit_r = 1$ ) und wartet anschließend darauf, daß sich  $r'$  durch eine Abänderung des Wertes von  $zeittestbit_r$  nach 0 zum Zeittest bereitmeldet. Dieses Warten dauert maximal ca.  $uhrparameter1$  Zeiteinheiten (Beschreibung der drei "Uhrparameter" s. 2.6). Trifft bis dorthin die Bereiterklärung nicht ein, so ist  $r'$  vermutlich im Rechnerhalt;  $r$  setzt dann selbst



zeittestbit<sub>r</sub> := 0<sup>1)</sup>, hebt die Zeitabgleichssperre auf und verläßt das Zeitpflegeprogramm 1. Der Zeitabgleich ist dadurch bis zur nächsten Weckeralarmbehandlung (die auch auf r' anfallen kann) vertagt.

#### 2.5.4 Zeittest

Meldet sich r' jedoch zum Zeittest bereit (r' befindet sich dann im Zeitpflegeprogramm 2, einem Teil der Rechnerkernalarmbehandlung, und ist durch die Bereiterklärung Zeitgehilfe geworden), so führt der Zeitmeister eine Languhrberechnung durch und signalisiert sofort nach dem Ablesen seiner Uhr (Ergebnis:  $u_1$ ) durch "zasignal := 1" (Signal via Kernspeicher, s. Anhang 1) dem anderen Rechnerkern, ebenfalls eine Languhrberechnung durchzuführen und deren Ende durch "zasignal := 0" zurückzumelden. Nach Eintreffen dieser Rückmeldung liest der Zeitmeister nochmals seine Uhr ab (Ergebnis:  $u_2$ ), um sicherzustellen, daß der Zeittest wirklich innerhalb kürzester Zeit erfolgt ist. Dazu muß  $u_2 \equiv u_1 + p \pmod{2^{24}}$  mit  $0 \leq p \leq \text{uhrparameter2}$  gelten<sup>2)</sup>. Andernfalls ist der Zeittest (z.B. wegen Verzögerung durch Rechnerhalt) unbrauchbar und der Zeitabgleich wird ebenfalls bis zur nächsten Weckeralarmbehandlung vertagt. (Das Zeitpflegeprogramm 1 wird wie in 2.5.7 verlassen).

./ 9

- 1) Mit dem Befehl BL. War der Wert von zeittestbit<sub>r</sub> unmittelbar vor dieser Wertzuweisung bereits 0, so hat sich r' doch noch als Zeitgehilfe bereit erklärt.
- 2) Der Fall, daß die Dauer zwischen den beiden Uhrabfragen des Zeitmeisters gerade um  $n \cdot 2^{24} + p$  ( $n=1,2,3,\dots$ ) Zeiteinheiten verzögert wird, ist so unwahrscheinlich, daß er ausgeschlossen bleiben darf.

./ 9

### 2.5.5 Nachholung einseitig versäumter Uhrumläufe

Ist der Zeittest als brauchbar erkannt worden, so werden die Ergebnisse der beiden Rechnerkerne miteinander verglichen.

Es sei

$$s'_0 = \text{languhrprobe}_0 + \text{languhraddend}_0,$$

$$s'_1 = \text{languhrprobe}_1 + \text{languhraddend}_1.$$

Es gibt nun ein ganzzahliges  $h$  (meistens  $h=0$ ) derart, daß gilt:

$$|s'_1 - s'_0 + h \cdot 2^{24}| \ll 2^{24}.$$

Dies sieht man folgendermaßen ein:

Die Languhraddenden wurden seit dem letzten Zeitabgleich nicht mehr verändert; damals waren sie so korrigiert worden, daß, wären die verglichenen Zeiten damals bereits mit den korrigierten Languhraddenden berechnet worden, die Zeitberechnung auf beiden Rechnerkernen dasselbe Ergebnis gebracht hätte. Unterstellt man, daß dieser Zeitabgleich nicht mehr als einen Tag zurückliegt, so ergibt sich, daß sich der Abstand der beiden Languhren voneinander seither um maximal  $2 \cdot 24 \cdot 60 \cdot 60$  verändert haben kann. Damit können sich aber auch  $s'_0$  und  $s'_1$  (die zugehörigen Uhrabfragen erfolgen ja nahezu gleichzeitig), von Vielfachen von  $2^{24}$  abgesehen, um nicht viel mehr als  $2 \cdot 24 \cdot 60 \cdot 60$  ( $\ll 2^{24}$ ) unterscheiden.

$|h|$  ist die Anzahl der Uhrumläufe, die seit dem letzten Zeitabgleich von dem einen Rechnerkern  $r_1$  gegenüber dem Rechnerkern  $r_2$  zusätzlich verpaßt wurden (erkenntlich an  $s'_1 \leq s'_0$ ); durch "languhrprobe <sub>$r_1$</sub>  +=  $h \cdot 2^{24}$ " wird die Anzahl der  $r_1$  seit  $r_2$  der Zeitfundierung von beiden Rechnerkernen verpaßten Uhrumläufe wieder gleich gemacht.

(Die Anzahl der seit dem letzten Zeitabgleich auf beiden Rechnerkernen gleichviel versäumten Uhrumläufe läßt sich innerhalb der Schicht 1 des BSM leider nicht feststellen.)

2.5.6 Zeitabgleich

Als Sollwert für die Systemzeit im Abgleichszeitpunkt wird nun bestimmt

$$s' := \text{zeitbasis} + \left[ \frac{\text{languhrprobe}_0 + \text{languhrprobe}_1}{2} \right]$$

und die Languhraddenden werden so neu besetzt, daß für  $q=0,1$  gilt:

$$s' = \text{languhrprobe}_q + \text{languhraddend}_q.$$

Durch

$$\text{zatermin} := s' + \text{uhrparameter3}$$

wird der Termin für den nächsten Zeitabgleich eingestellt.

2.5.7 Beendigung der Zeitpflegearbeiten

Rechnerkern r' wird durch "zasignal := 1" aufgefordert, die Zeitgehilfenschaft zu beenden; das entsprechende Rücksignal "zasignal = 0" wird von Rechnerkern r abgewartet. Dadurch ist sichergestellt, daß ein Zeitintervall, in dem ein Rechnerkern Zeitgehilfe ist, stets überdeckt wird von einem Zeitintervall, in dem der andere Rechnerkern Zeitmeister ist.

Nach einem Zeitabgleich verläßt ein Rechnerkern das Zeitpflegeprogramm 1 bzw. 2 erst, nachdem er festgestellt hat, daß die mit dem neuen Languhraddenden berechnete Systemzeit  $\geq \max(s'_0, s'_1)$  ist. Damit wird die Monotonie der Ergebnisse von Systemzeitberechnungen über Zeitabgleiche hinweg garantiert.<sup>1)</sup>

---

1) Zunächst war vorgesehen, beim Zeitabgleich die Zeitrechnung desjenigen Rechnerkerns, der beim Zeittest das kleinere Ergebnis lieferte, durch Erhöhung des Languhraddenden an die des anderen Rechnerkerns anzupassen. Dadurch wäre die angestrebte Monotonie von selbst gegeben, jedoch wäre die Definition der Systemzeit nicht mehr so einfach. Hier wurde eine einfache Definition einem einfachen Programm vorgezogen, zumal der dynamische Mehraufwand nur etwa 1o Befehle pro sec. beträgt.



Vor dem Verlassen des Zeitpflegeprogramms 1 wird noch die Zeitabgleichssperre aufgehoben.

## 2.6 Die drei "Uhrparameter"

In die Zeitpflege gehen 3 konstante Parameter ein, deren Wert hier vorläufig festgelegt ist und später, nach Beobachtung der bei der Zeitpflege anfallenden Meßgrößen, evtl. noch zweckmäßiger gewählt werden kann.

### 2.6.1 uhrparameter1 = 15

Soviele Zeiteinheiten wird vom Zeitmeister nach Absendung eines Rechnerkernalarms maximal auf die Bereitmeldung des anderen Rechnerkerns zum Zeittest gewartet. In diesem Zeitraum soll der andere Rechnerkern, wenn er nicht gerade im Rechnerhalt ist, immer in das Zeitpflegeprogramm 2 (Teil der Rechnerkernalarmbehandlung) gelangen können, auch wenn er bei Eintreffen des Rechnerkernalarms zufällig noch unter Alarmsperre 1 laufen sollte.

### 2.6.2 uhrparameter2 = 3

Der Zeitmeister fragt beim Zeittest zweimal seine Uhr ab; einmal vor und einmal nach der Languhrberechnung des Zeitgehilfen. Der Zeittest gilt nur dann als ungestört und brauchbar, wenn die Uhr des Zeitmeisters zwischen den beiden Uhrabfragen höchstens uhrparameter2 mal hochzählt. Mit uhrparameter2 = 3 darf als gesichert angesehen werden, daß die beiden Zeitpunkte  $t_0$  und  $t_1$ , in denen Rechnerkern 0 und 1 ihre Uhren für den Zeittest abfragen, weniger als 9,9999  $\mu$ s auseinanderliegen; dies wurde für die Genauigkeitsaussage in 2.2 bereits verwendet.

### 2.6.3 uhrparameter3 = 84000

Minimale Zeitspanne, die zwischen zwei aufeinanderfolgenden Zeitabgleichen liegen soll.

Die Wahl von  $\text{uhrparameter3} = 84000$  wird folgenden Forderungen gerecht:

1. Zeitabgleiche sollen nicht häufiger stattfinden als nötig.
2. Folgen in einer Bearbeitung zwei Systemzeitberechnungen  $S_1$  und  $S_2$  unmittelbar aufeinander (Ergebnisse:  $s_1$  und  $s_2$ ), so soll stets  $s_2 \geq s_1$  gelten.

Die zweite Forderung soll etwas genauer beleuchtet werden:

Liegt zwischen  $S_1$  und  $S_2$  ein Zeitabgleich, so ist  $s_2 \geq s_1$  nach 2.5.7 gewährleistet. Andernfalls ist, wenn  $S_1$  und  $S_2$  auf demselben Rechnerkern ablaufen, durch die Berechnungsformel aus 2.2 ebenfalls  $s_2 \geq s_1$  gesichert, weil sich die Languhraddenden zwischen Zeitabgleichen nicht ändern. Nur wenn  $S_1$  und  $S_2$  zwischen denselben Zeitabgleichen, aber auf verschiedenen Rechnerkernen ausgeführt werden, ist  $s_2 < s_1$  zu befürchten. Findet  $S_1$  im (absoluten) Zeitpunkt  $t_1$ ,  $S_2$  in  $t_2$  statt ("Zeitpunkt der Berechnung" = Zeitpunkt der zugehörigen Uhrabfrage), so wird  $t_2 \geq t_1 + 55 \mu\text{s}$  unterstellt. (Zwischen  $S_1$  und  $S_2$  liegt zumindest die Losbindung vom dem einen Rechnerkern und ein großer Teil der Rechnerkernvergabe auf dem anderen.) Dann läßt sich zeigen (Unterlagen sind beim Verfasser), daß  $s_2 \geq s_1$  gewährleistet ist, wenn zwei Uhrabgleiche nie mehr als 1,5 sec auseinanderliegen. Da in jeder Weckeralarmbehandlung geprüft wird, ob ein Zeitabgleich fällig ist, und spätestens ca. 0,66 sec nach dem Zeitabgleichstermin eine Weckeralarmbehandlung anfallen dürfte, wird  $\text{uhrparameter 3}$  entsprechend  $(1,5 - 0,66)$  sec gewählt.

## 2.7 Brauchbarkeit der simulierten Systemuhr

Z.Z. läßt sich noch nicht ganz übersehen, an welchen Stellen sich die Differenz zwischen berechneter und definierter Systemzeit unangenehm bemerkbar macht.

Erstrebenswert erscheint eine "brauchbare Systemuhrsimulation": Aus der Sicht eines Bearbeiters sollte eine Zeitabfrage denselben Effekt haben, wie wenn es ein für beide Rechnerkerne einheitliches Languhrregister gäbe, die "Systemuhr", die nach jeweils einer Zeiteinheit um 1 hochzählt. Bei einer "brauchbaren Systemuhrsimulation" soll vor allem die Relation

(1\*)  $\text{Bearbeiterfortschritt} \leq \left( \frac{\text{vom Bearbeiter festgestellter Fortschritt}}{\text{Fortschritt der absoluten Zeit}} \right)$   
erfüllt sein, d.h. genauer:

Frägt ein Bearbeiter zweimal die Systemuhr ab, zuerst im (absoluten) Zeitpunkt  $t_1$  mit dem Ergebnis  $s_1$ , später im Zeitpunkt  $t_2$  mit dem Ergebnis  $s_2$ , und dauern die Phasen, in denen der Bearbeiter in der Zwischenzeit einen Rechnerkern hat ("Rechnerkernphasen"), zusammen-  
genommen  $\Delta b \leq (t_2 - t_1)$ , so soll in jedem Fall

$$\left[ \frac{\Delta b}{10,0001 \mu s} \right] \leq s_2 - s_1$$

gelten. (10,0001  $\mu s$ : Maximale Dauer einer Zeiteinheit) Eine Folge von (1\*) ist insbesondere, daß in jedem Fall die "Monotonie der Systemuhr für einen Bearbeiter", nämlich

$$(2*) \quad s_2 \geq s_1$$

erfüllt ist.

Andererseits ist aber das Erfülltsein von (2\*) auch schon hinreichend für das von (1\*):

Während einer Rechnerkernphase  $P_1$  hat der Bearbeiter B immer denselben Rechnerkern  $r$  und die für ihn simulierte Systemzeit wird nach 2.2 aus dessen Languhr und einem während  $P_1$  konstanten Languhraddenden berechnet. (1\*) ist mithin während einer Rechnerkernphase erfüllt.



Ergibt nun die für B in der nächsten Rechnerkernphase  $P_2$  simulierte Systemzeit in jeden Fall bereits zu Beginn von  $P_2$  einen Wert, der größer oder gleich dem ist, der für das Ende von  $P_1$  maßgeblich war, (was äquivalent ist damit, daß  $(2^*)$  in jedem Fall erfüllt ist) so ist die Relation  $(1^*)$  für den Bearbeiter B immer erfüllt.

### 2.8 Auswirkung von Rechnerhalten auf die Zeitverfolgung

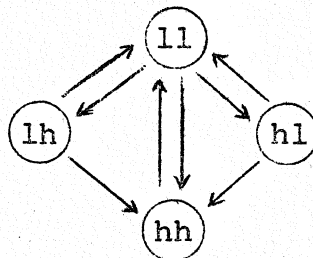
Im folgenden werden für jeden Rechnerkern unterschieden die Zustände l (laufend, d.h. Taste "HALT" des Bedienpultes ist für den Rechnerkern nicht eingeschaltet) und h (haltend, d.h. Taste "HALT" ist eingeschaltet). Durch Hintereinanderschreiben der Zustände von Rechnerkern 0 und 1 möge sich der Zustand der Anlage ergeben:

ll (beide Rechnerkerne laufen),

lh oder hl (einseitiger Rechnerhalt),

hh (zweiseitiger Rechnerhalt).

Bei den vorliegenden Zeitpflegeprogrammen sind nur solche Rechnerhalte eingeplant, mit deren Beendigung stets ein Übergang nach ll verbunden ist. (Andere Rechnerhalte kommen vermutlich in der Praxis nicht vor. Falls doch, so müßten die Zeitpflegeprogramme, soweit das möglich ist, auch noch auf diese Fälle abgestimmt werden.) Die eingeplanten Rechnerhalte sind gerade die, die mit folgendem Zustandsübergangsdiagramm im Einklang stehen:



Treten nur eingeplante Rechnerhalte auf, so kann man erwarten, daß die hier angegebenen Zeitpflegeprogramme eine brauchbare Systemuhrsimulation im Sinne von 2.7 liefern. Dies soll durch die folgenden Punkte 2.8.1 bis 2.8.3 begründet werden.

#### 2.8.1 Einseitiger Rechnerhalt

Hält nur ein Rechnerkern, so wird die Zeit auf dem laufenden Rechnerkern weiterverfolgt, indem dessen Languhrprobe zumindest in jeder Weckeralarmbehandlung wieder auf den neuesten Stand gebracht wird. Zeitabgleiche sind jedoch nicht mehr möglich. Die Abweichung der berechneten Systemzeit von der definierten kann dabei zunehmen (max. 1 Zeiteinheit pro sec), doch dürfte dies wohl nirgends stören.

#### 2.8.2 Zweiseitiger Rechnerhalt

Halten beide Rechnerkerne gleichzeitig, so kann dabei die Zeit nicht mehr weiterverfolgt werden. Wie schon früher bemerkt, ist bei einem zweiseitigen Rechnerhalt von über 2,7 min Dauer damit zu rechnen, daß ein Vielfaches von Uhrumläufen bei der Verfolgung der Systemzeit verpaßt wird. (Auf eine eventuelle Nachholung dieser verpaßten Uhrumläufe mithilfe einer Schreibmaschineneingabe wird in dieser Schrift nicht eingegangen.)

#### 2.8.3 Brauchbarkeit der Systemuhrsimulation auch bei Rechnerhalten

Durch Rechnerkernhalte (von eingeplanten Typus) kann der Abstand zwischen zwei aufeinanderfolgenden Zeitabgleichen zwar über das normale Maximum von 1,5 sec hinaus verlängert werden, jedoch steht in diesem Falle dann bei Ende des Rechnerhalts (= Rückkehr in den Zustand 11) ein Weckeralarm an und ein Zeitabgleich ist fällig, der auch unverzüglich durchgeführt wird.<sup>1)</sup>

---

1) Dabei ist allerdings unterstellt, daß ein Zeitabgleichsversuch eines Rechnerkerns nur dann scheitern kann, wenn der andere Rechnerkern im Rechnerhalt ist

Daß die Brauchbarkeit der simulierten Systemuhr auch trotz dieser Rechnerhalte weiterbesteht, soll die folgende Überlegung zeigen: Liegt der Abstand der beiden aufeinanderfolgenden Zeitabgleiche, die im Zeitpunkt  $t_1$  und  $t_2$  stattfinden mögen, über 1,5 sec, so ist die "Monotonie der Systemuhr für einen Bearbeiter" in den ersten 1,5 sec nach 2.6.3 gewährleistet; in der Zeit von  $(t_1 + 1,5 \text{ sec})$  bis  $t_2$  kommt aber höchstens noch ein Rechnerkern für Rechnerkernphasen eines Bearbeiters in Betracht, weil der andere in dieser Zeit im Rechnerhalt ist und anschließend unter Alarmsperre 1 läuft (Teilnahme am Zeitabgleich). Damit ist aber die "Monotonie der Systemuhr für einen Bearbeiter" auch im Zeitintervall  $(t_1 + 1,5 \text{ sec}, t_2)$  gesichert, woraus nach 2.7 die Brauchbarkeit der Systemuhrsimulation im ganzen Zeitintervall  $(t_1, t_2)$  folgt.

## 2.9 Programmbeschreibungen

### 2.9.1 Zeitberechnungsprogramm

Zur Zeitberechnung ist das Unterprogramm SYSZEIT zu verwenden. Aufruf nur von Programmen aus, die in Großkachel 0 liegen und im Systemmodus laufen.

Aufruf: SFB SYSZEIT

Resultat<sup>1)</sup>: A Ø Ø ~~N~~ ~~M~~ Y Ø U K.

$\langle A \rangle = TK1, z.$

z ist die berechnete Systemzeit;  $1 \leq z < 2^{46}$ .

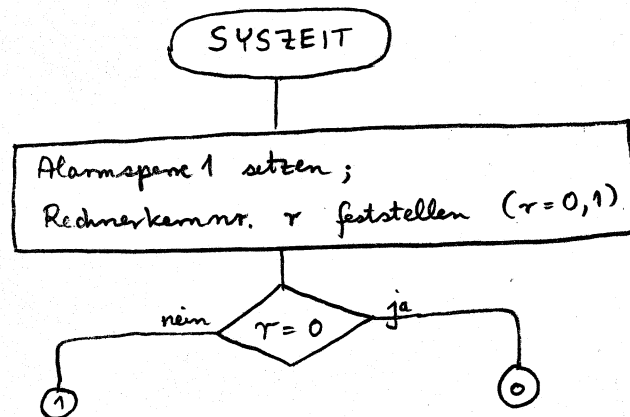
---

1) Bedeutung der Notation: Ist eine Registerbezeichnung (1) angegeben bzw. (2) durchstrichen angegeben bzw. (3) unterstrichen angegeben, so hat das zugehörige Register (1) denselben Inhalt wie vor Aufruf des Unterprogramms bzw. (2) undefinierten Inhalt bzw. (3) den explizit angegebenen Inhalt.

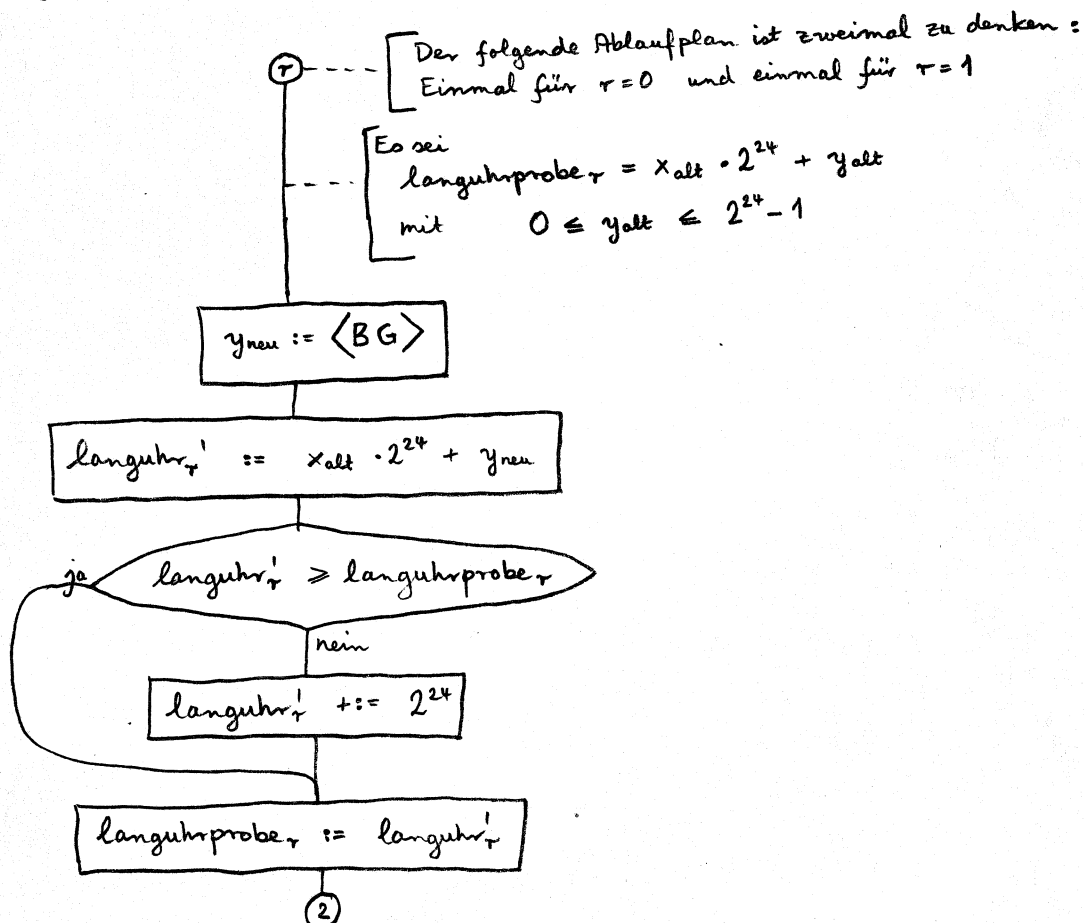


## 2.9.1.1 Ablaufplan zum Unterprogramm SYSZEIT

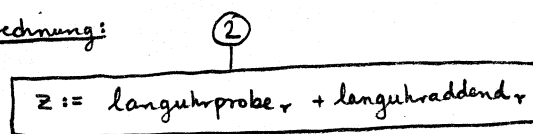
Vorspiel:



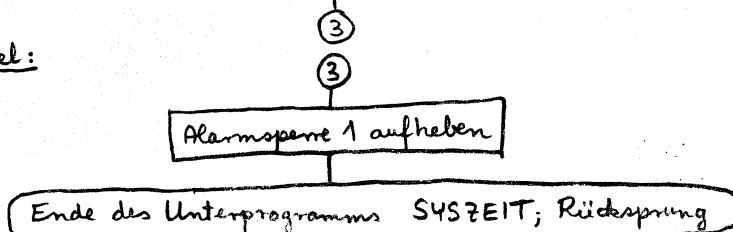
Languhrberechnung (= Anfang der Zeitberechnung):



Ende der Zeitberechnung:



Nachspiel:



2.9.1.2 TAS-Protokoll zum Unterprogramm SYSZEIT

```

, ,SYSZEIT.=R    BQ B      --(Q) = RUECKSPRUNGADRESSE--
,              BSS 'C0' '02' --BITS A9,A10,A23 = ALARMSPERRE 1--
,              --SETZEN UND KENNBITS NACH A BRINGEN--
,              ET ('2'/L3)  --BIT 3 (FUER RECHNERKERN 0 AUSBLEND.--
,              SNO ENDE0    --FALLS r=0--

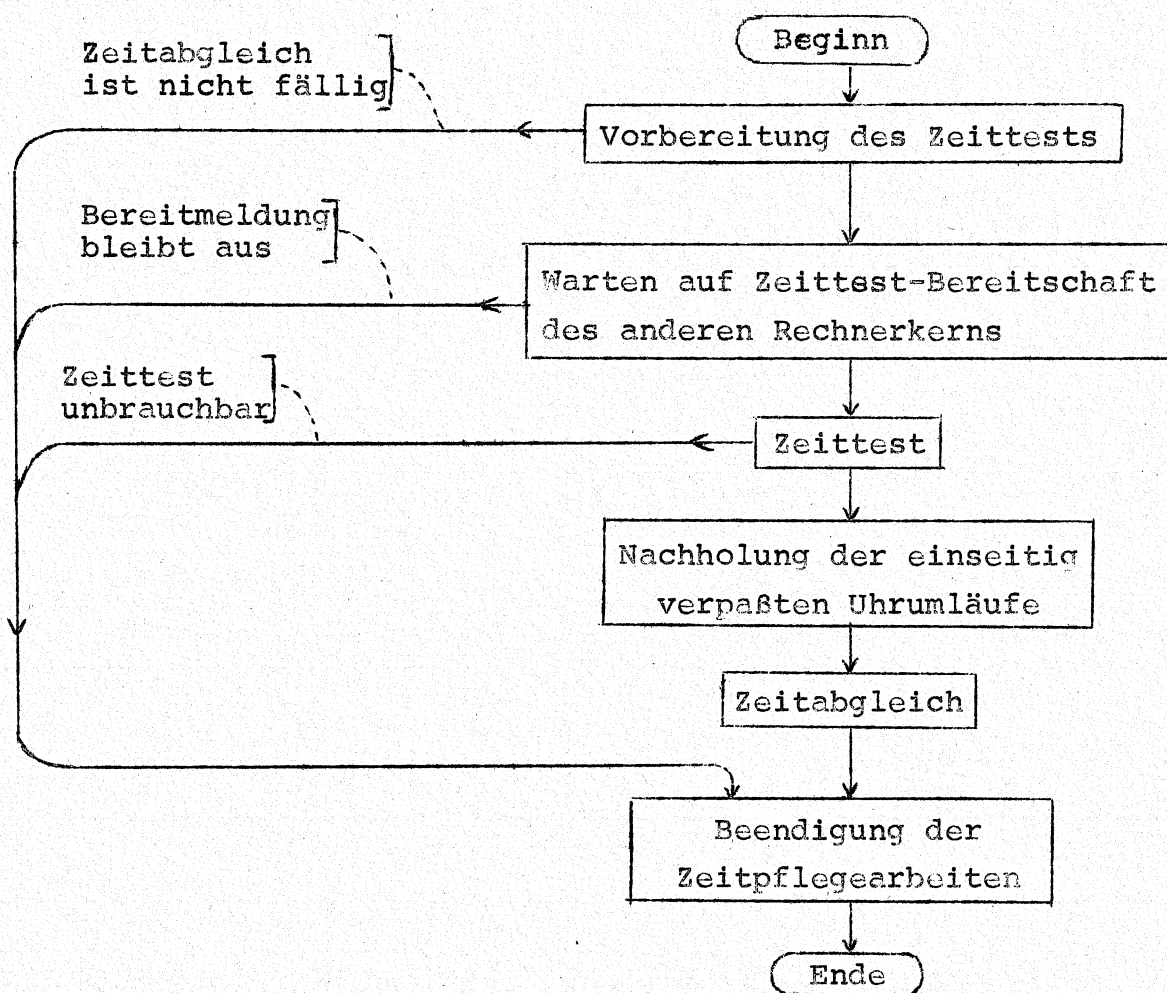
, *(ENDE=      VMØ 0 8      --UHRINHALT NACH B HOLEN--
,              R    B B
,              BH ('FF FFFF'/L1)
,              ZUS LANGUHRPROBE*(1)
,              RT    HD
,              SGG 2R
,              A    ('1 00 0000'/1)
,              C    LANGUHRPROBE*(1)
,              A    LANGUHRADDEND*(1)
,              BSS '40' '10' --BITS A10,A20 = ALARMSPERRE LOESCHEN--
,              R    SEQ)

,              *(ENDE,1)
, ENDE0=       *(ENDE,0)

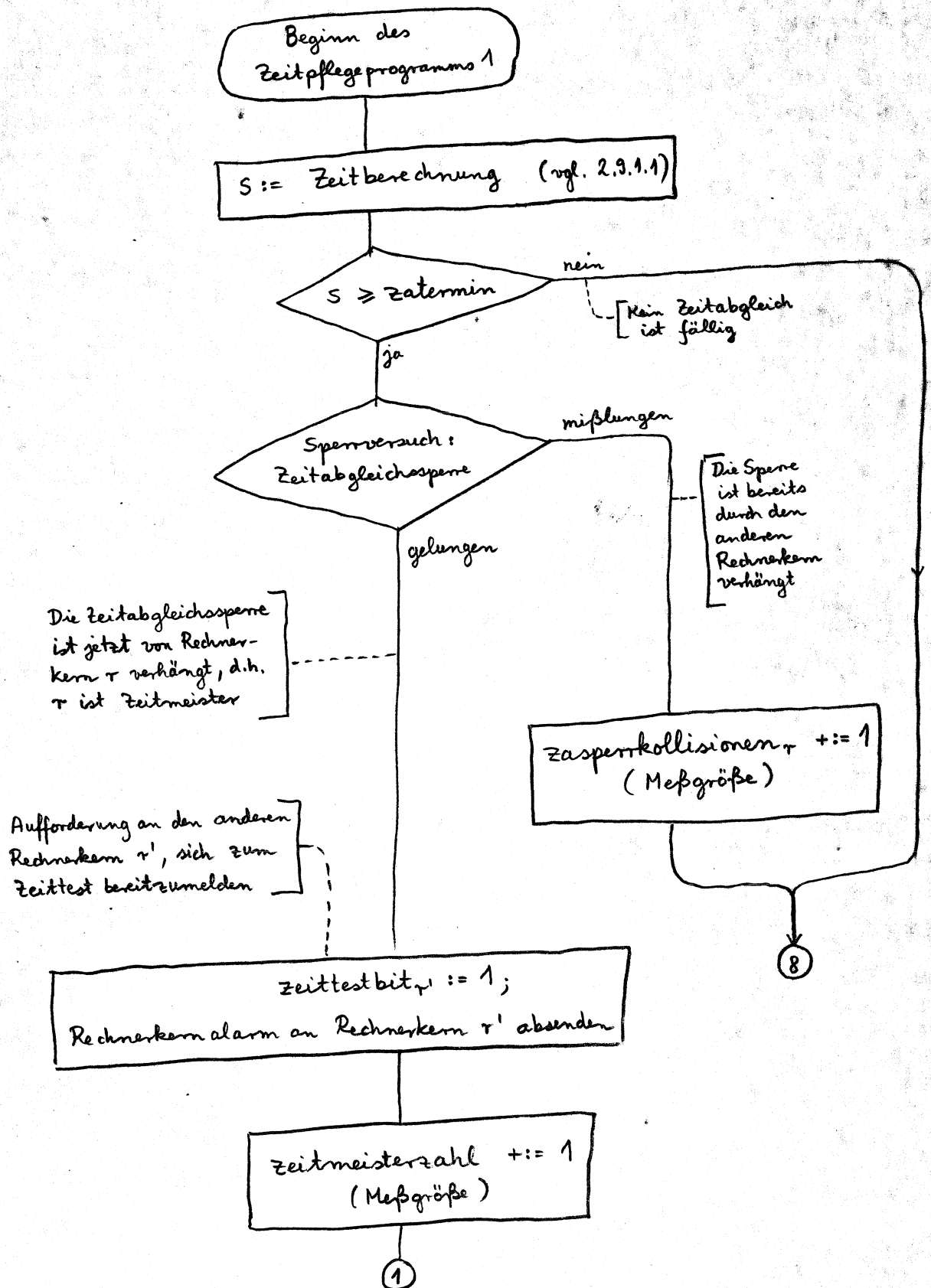
```

2.9.2 Zeitpflegeprogramm 1 (Teil der Weckeralarmbehandlung)

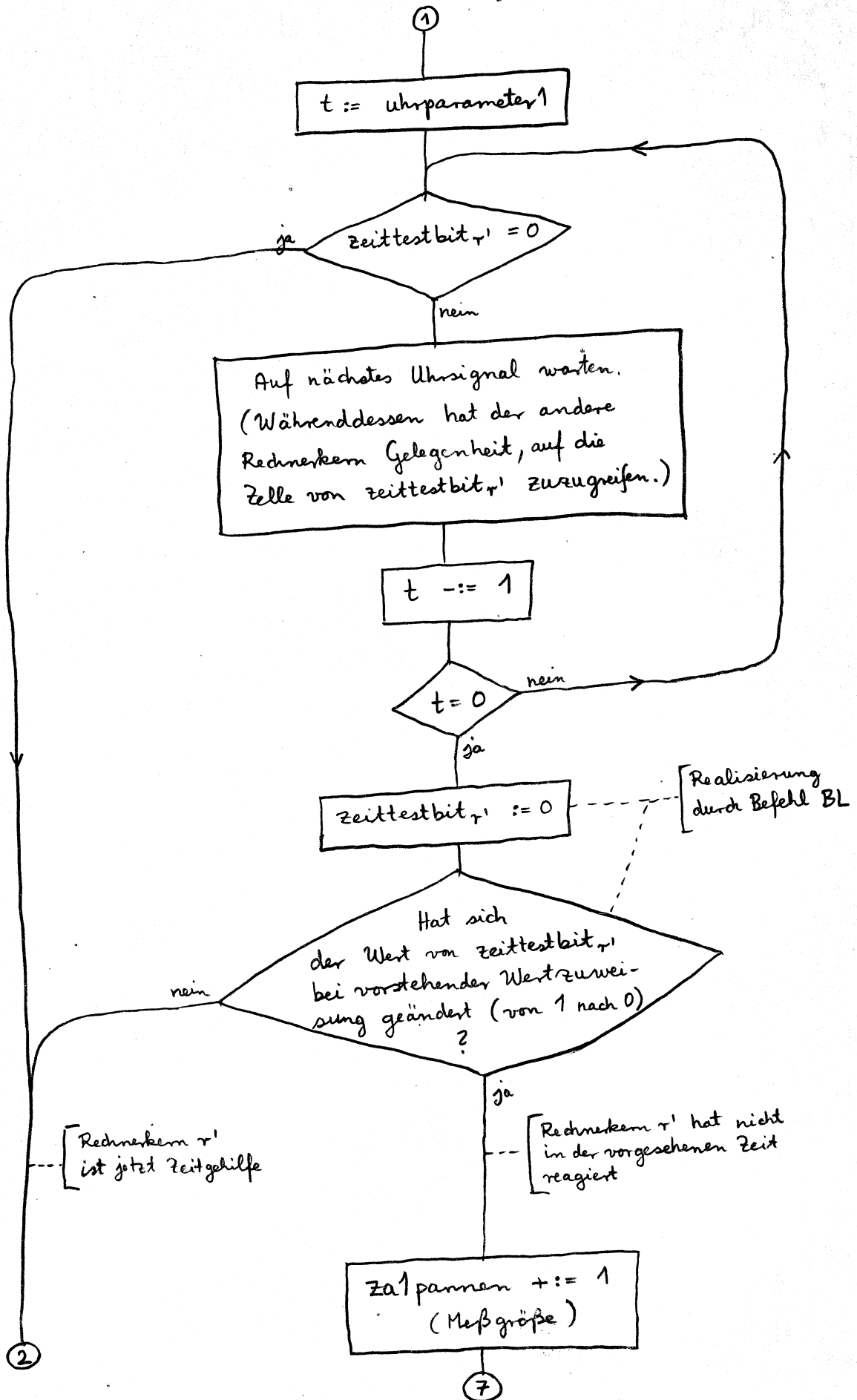
Das Zeitpflegeprogramm 1 (vgl. auch 2.5) wird bei jeder Weckeralarmbehandlung durchlaufen, durchwegs unter Alarmsperre 1.  $r=0,1$  sei die Nummer des ausführenden Rechnerkerns,  $r'$  die des anderen.

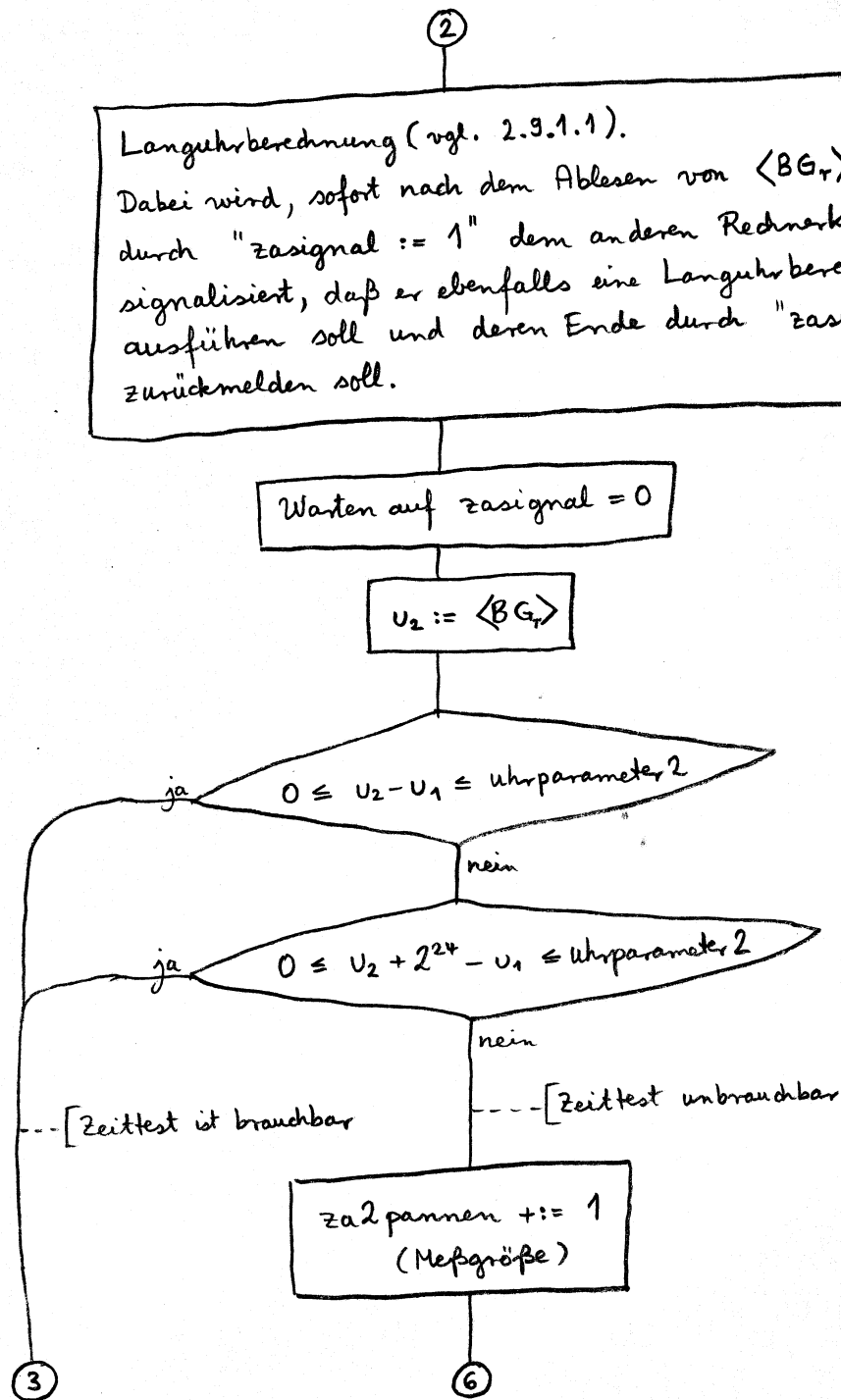
2.9.2.1 Grober Ablaufplan



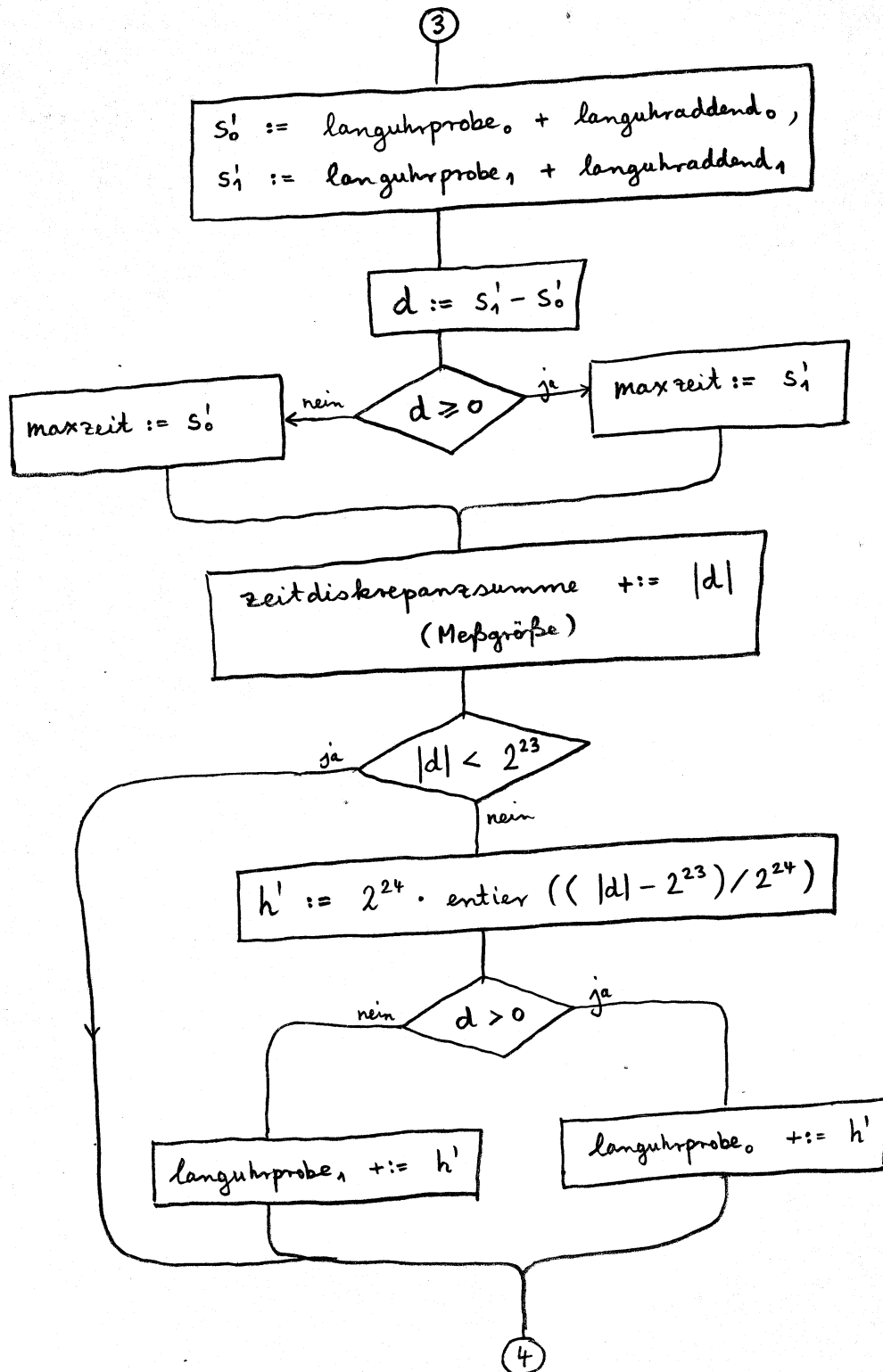
2.9.2.2 Detaillierte Ablaufpläne2.9.2.2.1 Vorbereitung des Zeittests

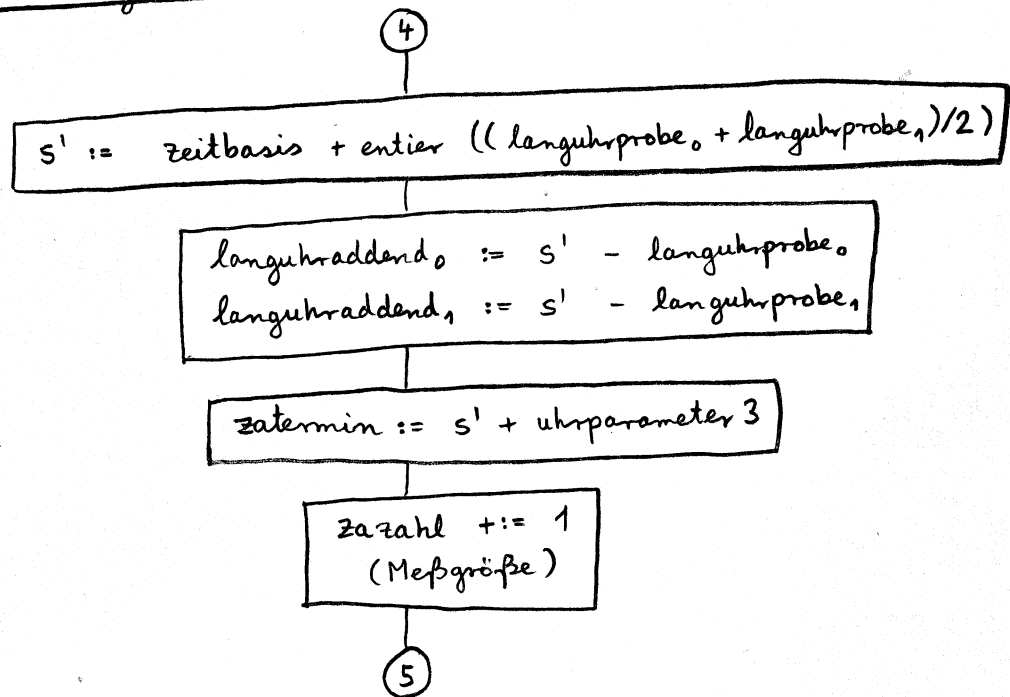
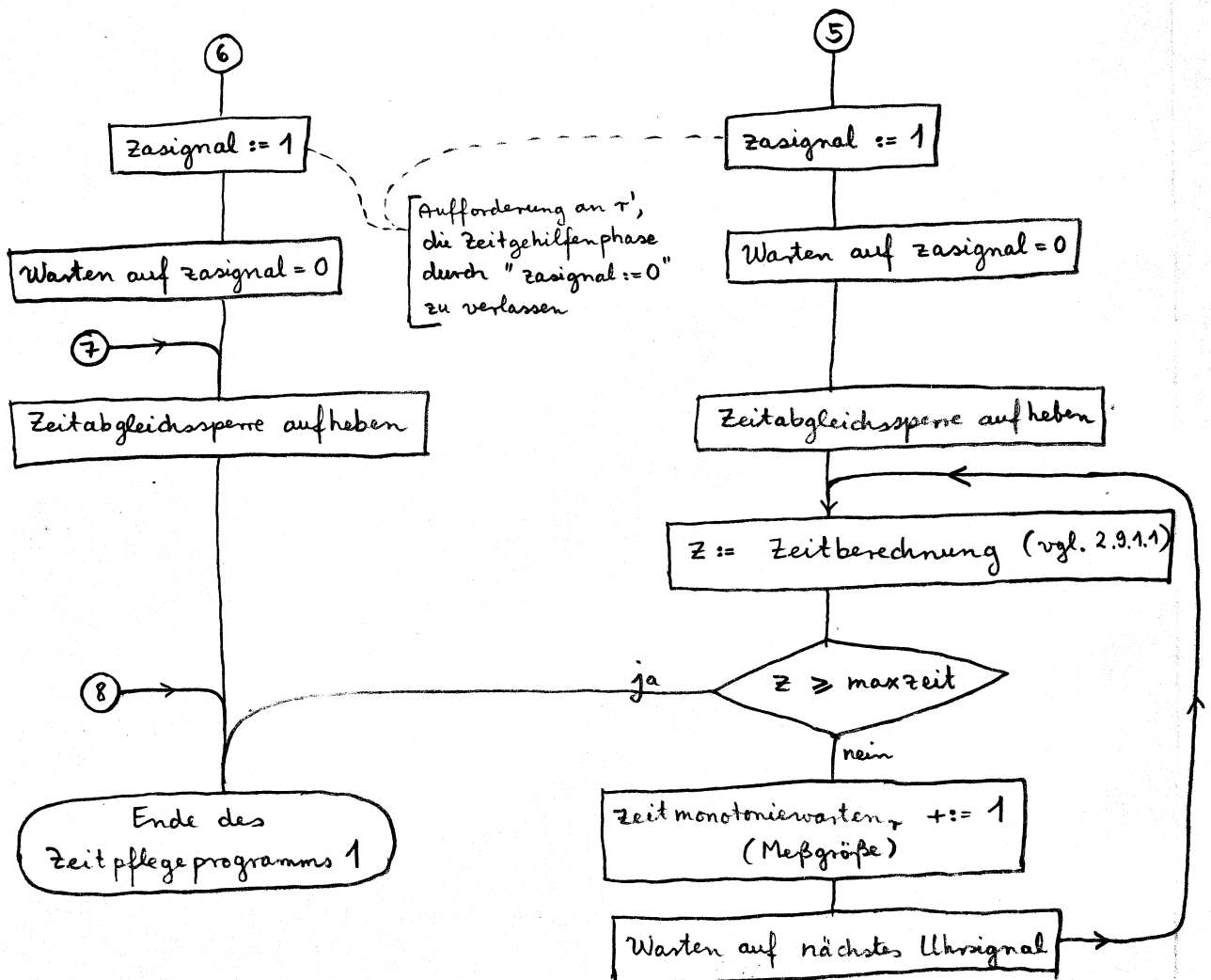
# 2.9.2.2.2 Warten auf Zeittest-Bereitschaft des anderen Rechnerkerns



2.9.2.2.3 Zeittest

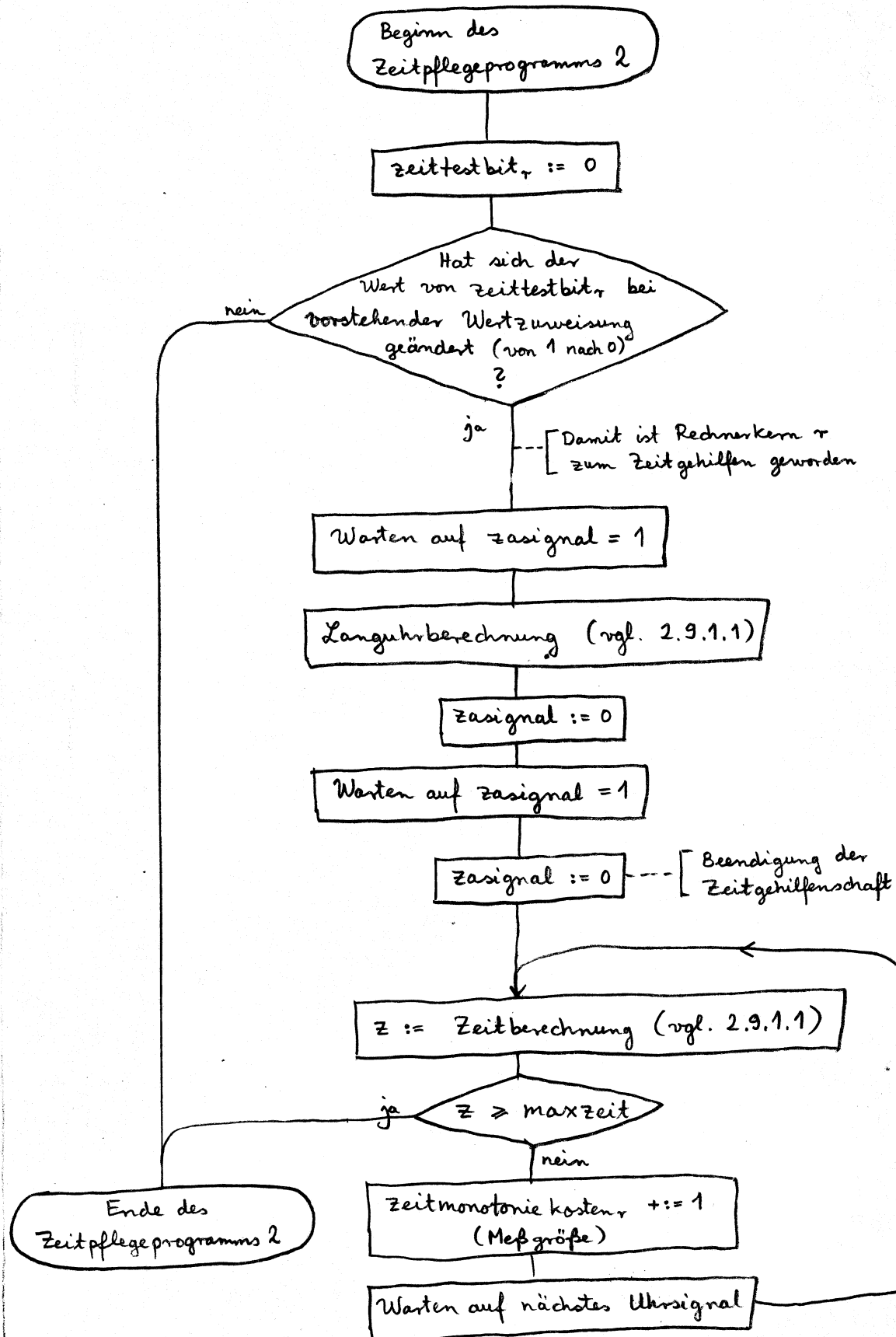


2.9.2.2.4 Nachholung der einseitig verpaßten Uhrumläufe

2.9.2.2.5 Zeitabgleich2.9.2.2.6 Beendigung der Zeitpflegearbeiten

### 2.9.3 Zeitpflegeprogramm 2 (Teil der Rechnerkernalarmbehandlung)

Das Zeitpflegeprogramm 2 (vgl. auch 2.5) wird bei jeder Rechnerkernalarmbehandlung durchlaufen, durchwegs unter Alarmsperre 1.  $r=0,1$  sei die Nummer des ausführenden Rechnerkerns,  $r'$  die des anderen.



### 3. Verbuchung von Rechnerkernzeiten

#### 3.1 Variable zur Messung des Verbrauchs an Rechnerkernzeit

Der Zeitmessung für einen Bearbeiter 1 dienen im Leitblock drei Größen (1 Einheit bedeutet 10  $\mu$ s):

$$(0 \leq \text{integer} \leq 2^{24} - 1) \text{ uhr1}_1$$

Stand des Uhrregisters bei der letzten Bindung des Bearbeiters 1 an einen Rechnerkern.

$$(0 \leq \text{integer} \leq 2^{24} - 1) \text{ uhr2}_1$$

Stand des Uhrregisters bei der letzten Lösung des Bearbeiters 1 von einem Rechnerkern.

$$(0 \leq \text{integer} \leq 2^{46} - 1) \text{ rekzeitverbrauch}_1$$

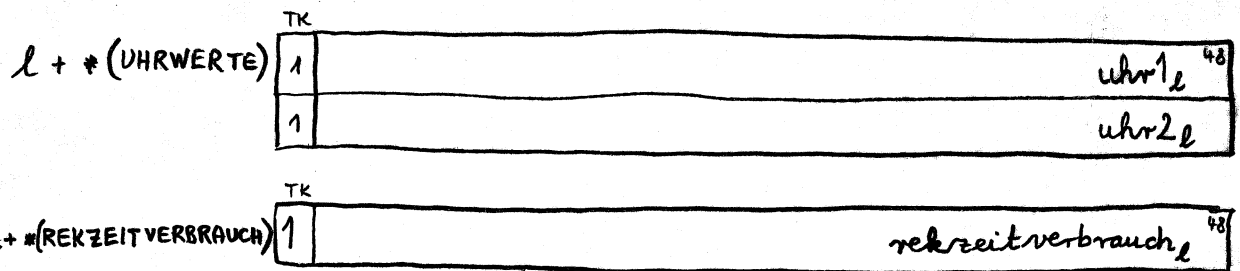
Rechnerkernzeit, die der Bearbeiter seit seiner Kreation verbraucht hat. Diese Größe wird bei jeder erneuten Bindung von Bearbeiter 1 an einen Rechnerkern um d Einheiten erhöht, entsprechend der Dauer der letzten Rechnerkernbenutzung:

```
d := uhr21 - uhr11;
if d < 0 then d += 224 fi;
if d ≥ 65535 then d := 65535 fi
```

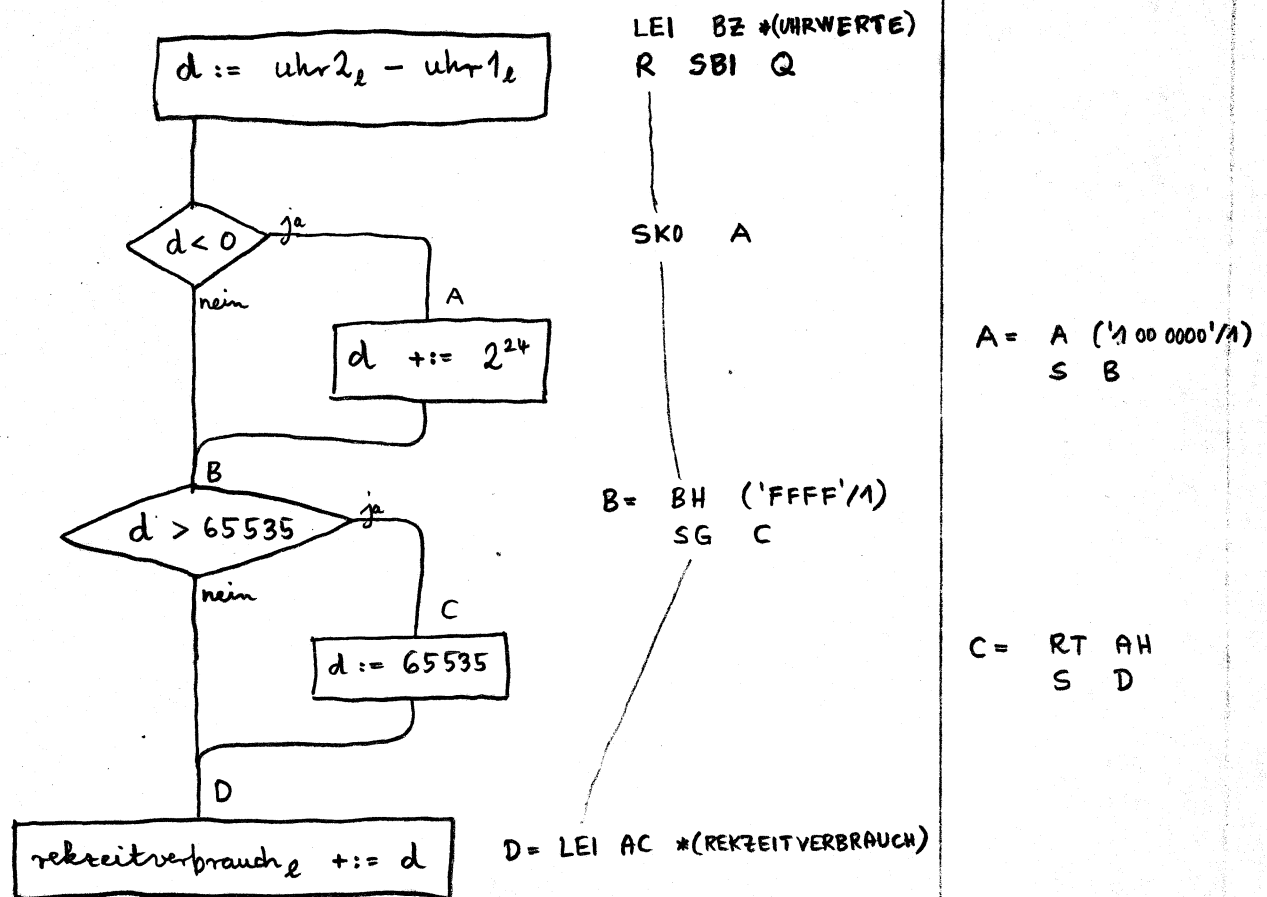
d liegt demnach stets im Bereich von 0 bis  $(2^{16} - 1)$ ; damit wird vermieden, daß ein in die Bindungsphase des Bearbeiters fallender Rechnerhalt dessen Rechnerkernzeitverbrauch grob verfälscht. (Im Normalbetrieb ist die Zeitspanne zwischen zwei aufeinanderfolgenden Weckeralarmabhandlungen auf einem Rechnerkern stets kleiner als  $2^{16}$ , mithin liegt auch die normale Dauer einer Bindungsphase unter  $2^{16}$  Zeiteinheiten.)



### Unterbringung der Größen im Leitblock:



### 3.2 Ablaufplan und TAS-Code zur Verbuchung der Rechnerkernzeit



Anhang 1. Zur Signalisierung zwischen Rechnerkernen via Kernspeicher

Eine Synchronisierung zwischen Zeitmeister und Zeitgehilfe kann über Rechnerkernalarme nicht erfolgen, weil beide Rechnerkerne unter Alarmsperre 1 laufen. (Sie wäre außerdem viel zu schwerfällig.) Sie erfolgt dadurch, daß der eine Rechnerkern den Inhalt einer Kernspeicherzelle ("Signalzelle") ändert und der andere Rechnerkern darauf wartet. Dabei darf es nicht vorkommen, daß dem einen Rechnerkern eine Signalabgabe unmöglich gemacht wird dadurch, daß der wartende Rechnerkern beim Speichervorrangwerk die höhere Priorität hat und pausenlos auf den Speichermodul mit der Signalzelle zugreift. Dies ist z.B. zu befürchten, wenn mit

X = B ZASIGNAL,

SNO X,

gewartet wird, weil auch der Befehl X in dem Speichermodul der Signalzelle liegen könnte.

Um auch dem signalabgebenden Rechnerkern die Zugriffsmöglichkeit auf alle Speichermoduln offenzuhalten, wird das Warten hier stets nach einem der folgenden beiden Schemata programmiert:

Schema 1:    X = WB 1,  
                  B . ZASIGNAL,  
                  SNO X,

Schema 2:    X = SH ZKU.96,  
                  B ZASIGNAL,  
                  SNO X,

Dabei führt der wartende Rechnerkern während der Ausführungsphase des Wartebefehls (hier ca. 8  $\mu$ s) bzw. des Sch iftbefehls (ca. 53 Takte  $> 2,5 \mu$ s) keine Speicherzugriffe aus und gibt damit dem anderen Rechnerkern Gelegenheit, auf einen gewünschten Speichermodul zuzugreifen. (Die Speicherzykluszeit ist 0,9  $\mu$ s.)

Schema 1 wird hier verwandt, wenn die Wartedauer durch ein Vielfaches von Zeiteinheiten begrenzt ist, Schema 2, wenn unbegrenzt gewartet wird. (Bei Schema 2 wird außerdem das Eintreffen des Signals meist schneller erkannt.)