WSP 432
Betriebsmikroprogramm

Zeichnungs-Nr.:
25.2201.600-00 BG

ÄNDERUNGSÜBERSICHT

Blatt 1

Register	1	2	3	4	5	6	7	8	9	10	Datum
	-	-	-	-	-						10.7.76
	-	A	-	-	-						6.4.77

Nähere Änderungshinweise siehe Änderungsleitblatt vor den einzelnen Registern

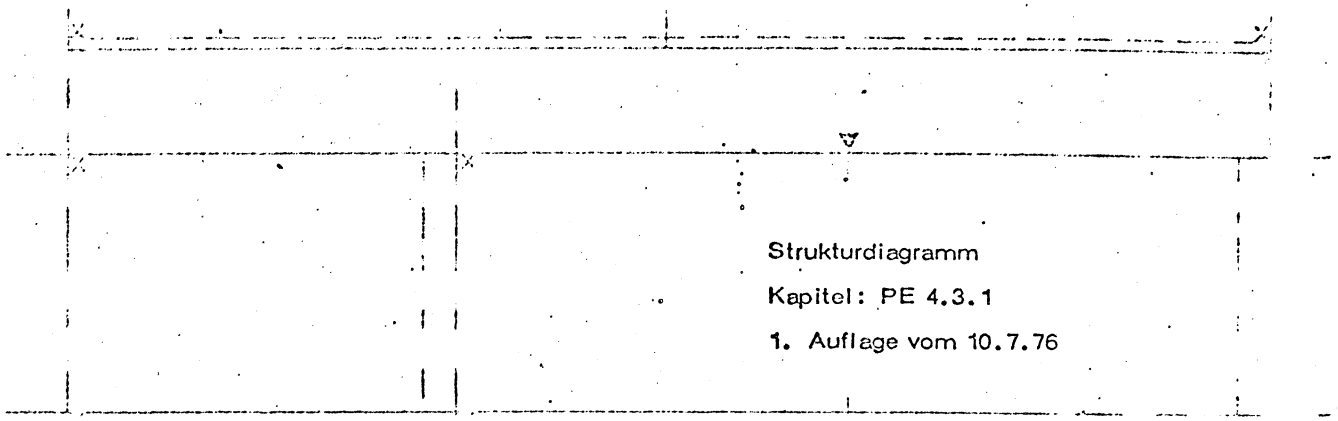
Betriebsmikroprogramm (BMP) WSP 432

TECHNISCHER KUNDENDIENST

WSP 432
Betriebsmikroprogramm
Strukturdiagramm, Kapitel PE 4.3.1

Zeichnungs-Nr.:
25.2201.600-00 BG

ÄNDERUNGSSTAND (ae)			Blatt 1
Reg. 1	ae	Auszutauschende bzw. zu ergänzende Seiten (*)	Datum
	-	-	10.7.76



Strukturdiagramm

Kapitel: PE 4.3.1

1. Auflage vom 10.7.76

umfaßt Blatt: PE 4.3.1-1 bis 4.3.1-460

ÄNDERUNGSSTAND (ae)			Blatt 1
Reg. 3	ae	Auszutauschende bzw. zu ergänzende Seiten (*)	Datum
	-	-	10.7.76

INHALTSVERZEICHNIS

1. ERLÄUTERUNGEN

- Die Abhandlung des Kanalverkehrs ist nur in der Startphase genauer dargestellt, da ansonsten die Übersichtlichkeit gelitten hätte.

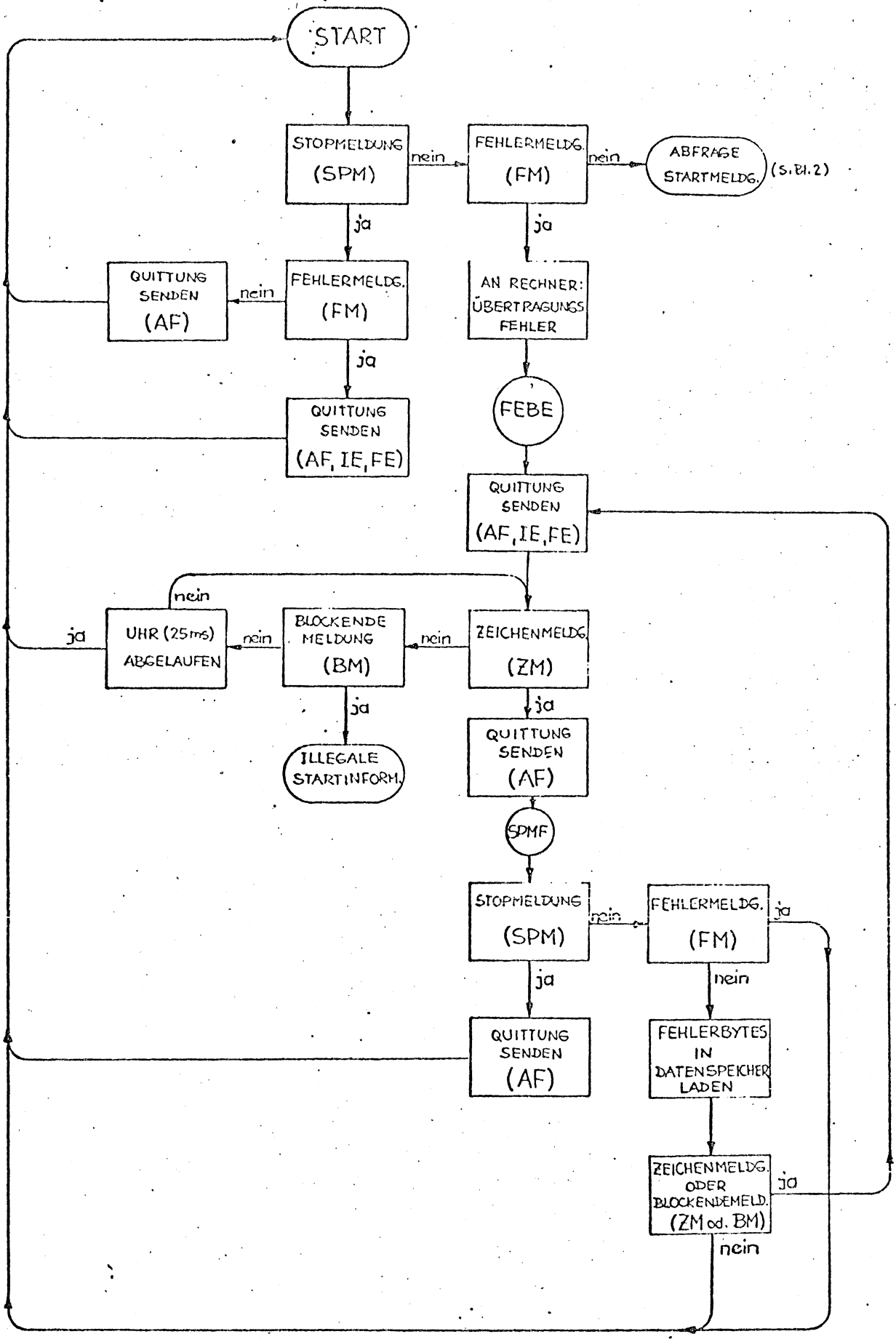
- Die Namen von Konnektoren (Marken) stimmen mit denen im Flußdiagramm überein (Ausnahme: Zahlen als Konnektoren).

- Im Ablauf der ECC-Korrektur wurde nur dargestellt, wo korrigiert und wie die Stelle zum Weiterlesen gefunden wird. Der genauere Ablauf ist von den Befehlen abhängig und am Schluß des Strukturdiagramms zu finden.

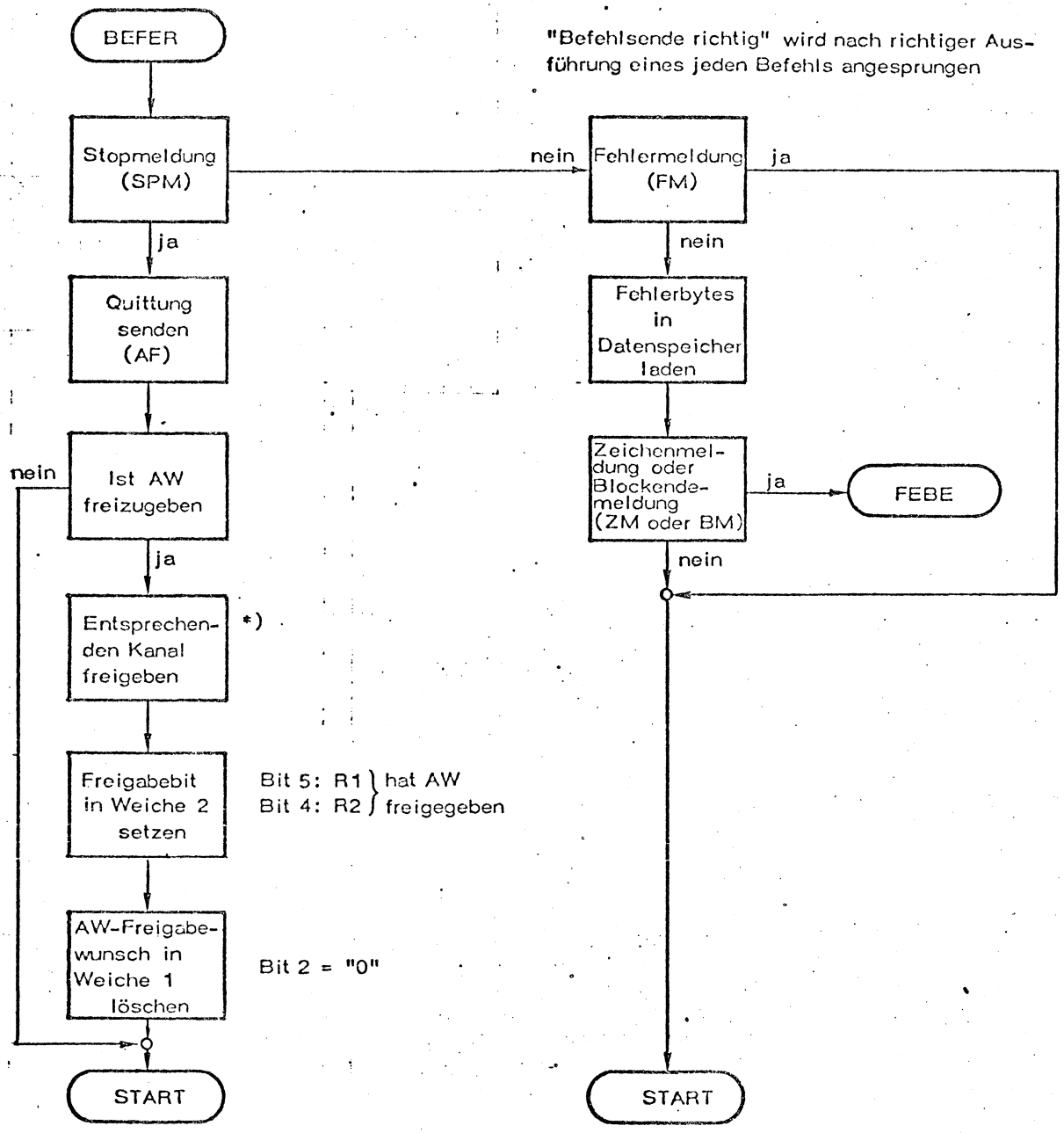
  Für eine genaue mathematische Einarbeitung in das ECC-Verfahren siehe Kapitel 4.3.4., "ECC-Verfahren".

- Alle Schleifen, die zu Endlosschleifen bei Nichteintreffen eines erwarteten Ereignisses werden können, sind durch die Uhr (MU) abgesichert, auch wenn dies der Übersichtlichkeit wegen nicht angegeben ist.

- In der Fehlerabhandlung werden die für das jeweilige Format nötigen Angaben in die Fehlerbytes geschrieben.

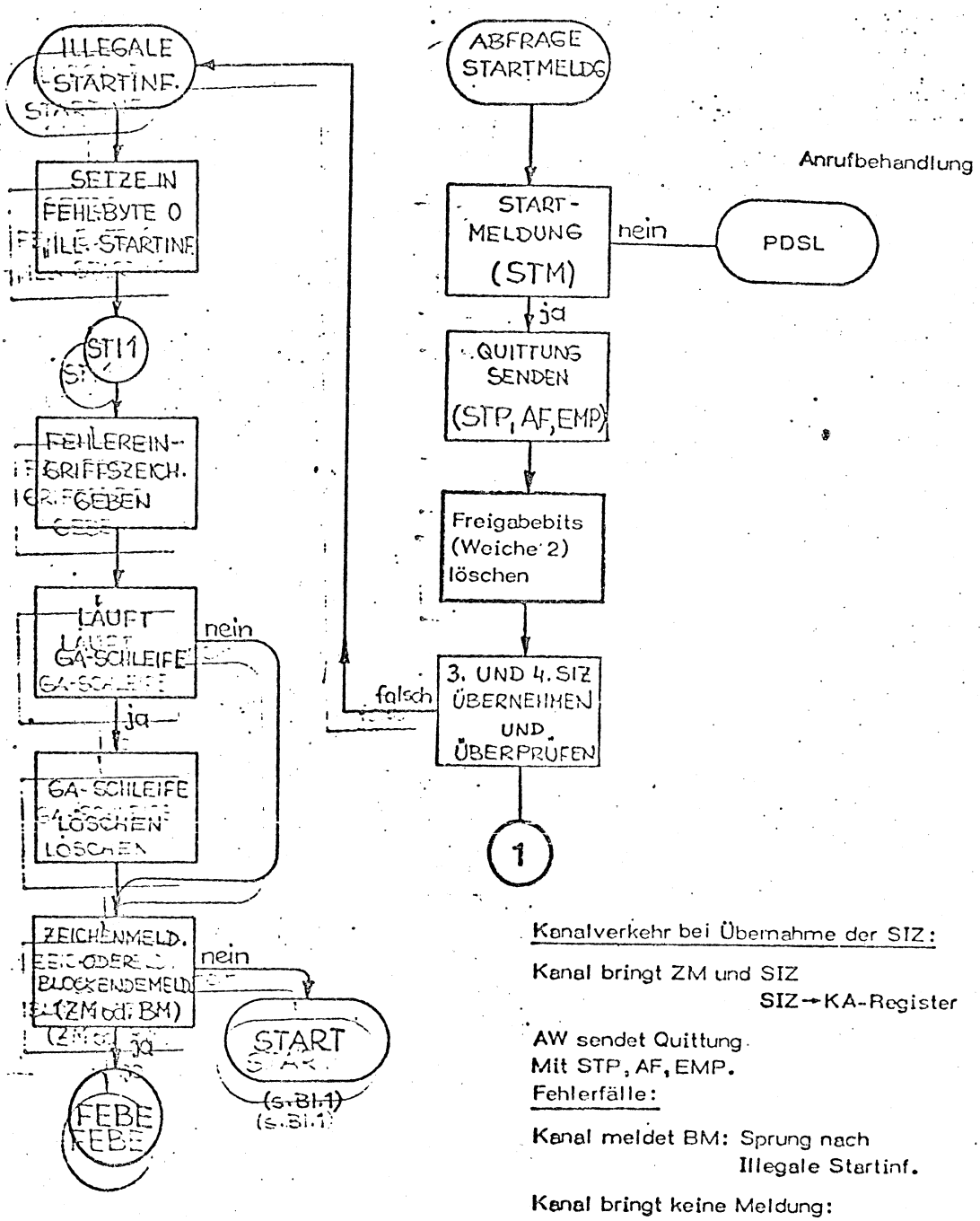


1.1. STARTPHASE (BL.1)



*) ST2 (R1) bzw. ST2 (2) = "L", nach 400 ns = "0"

1.1. ENDPHASE (Bl. 1a)



Kanalverkehr bei Übernahme der SIZ:

Kanal bringt ZM und SIZ
SIZ → KA-Register

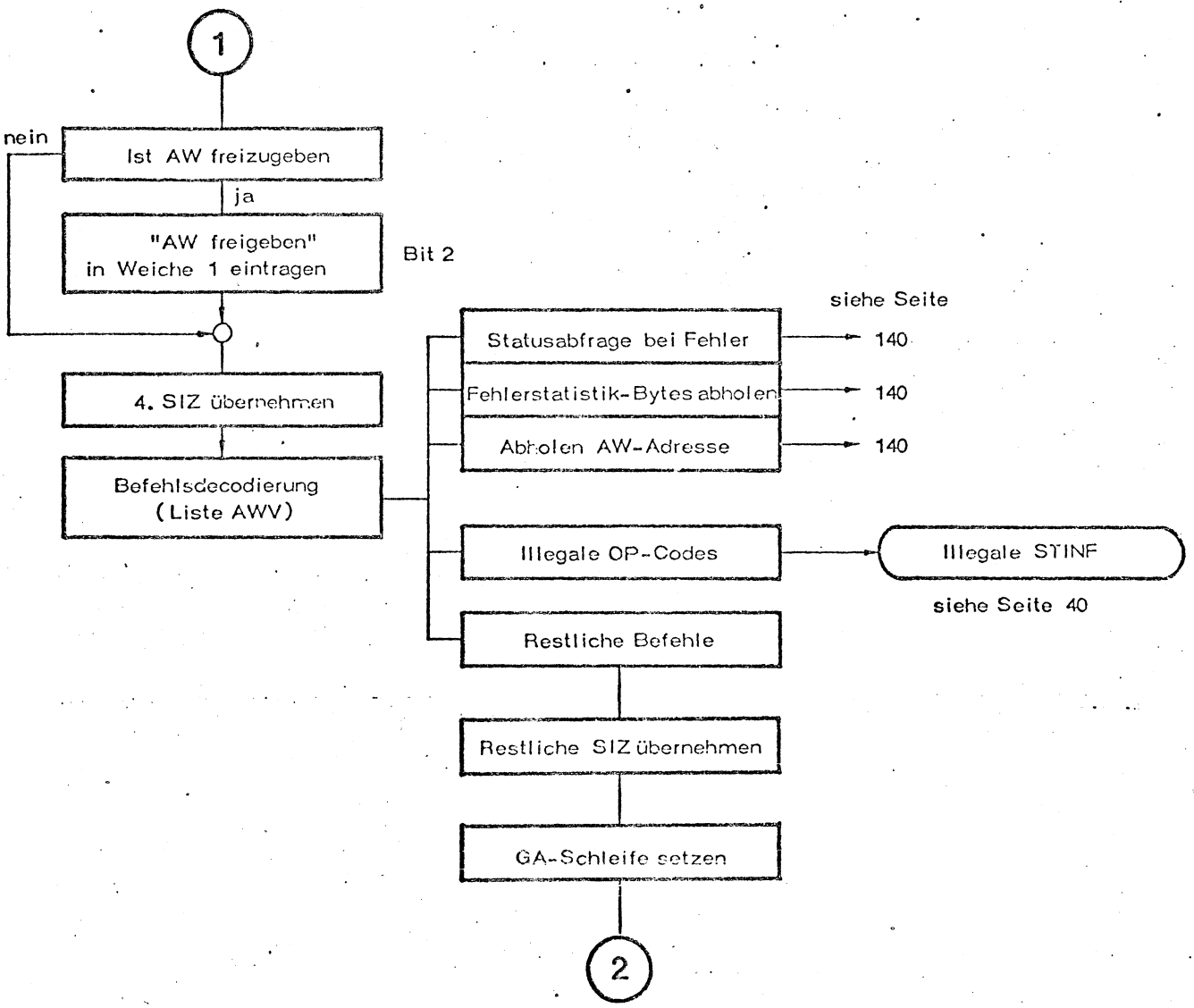
AW sendet Quittung
Mit STP, AF, EMP.

Fehlerfälle:

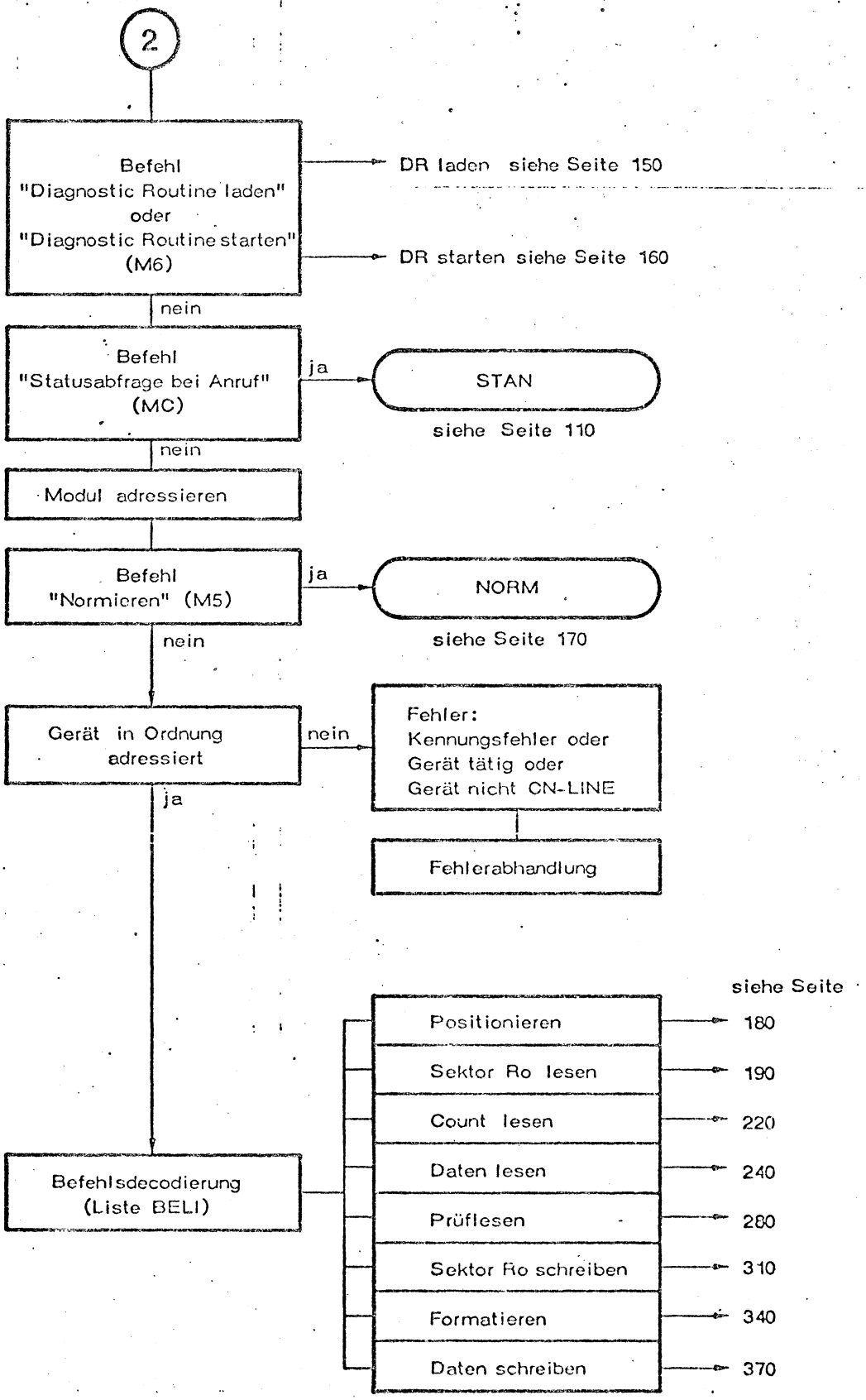
Kanal meldet BM: Sprung nach
Illegale Startinf.

Kanal bringt keine Meldung:
Sprung nach Start

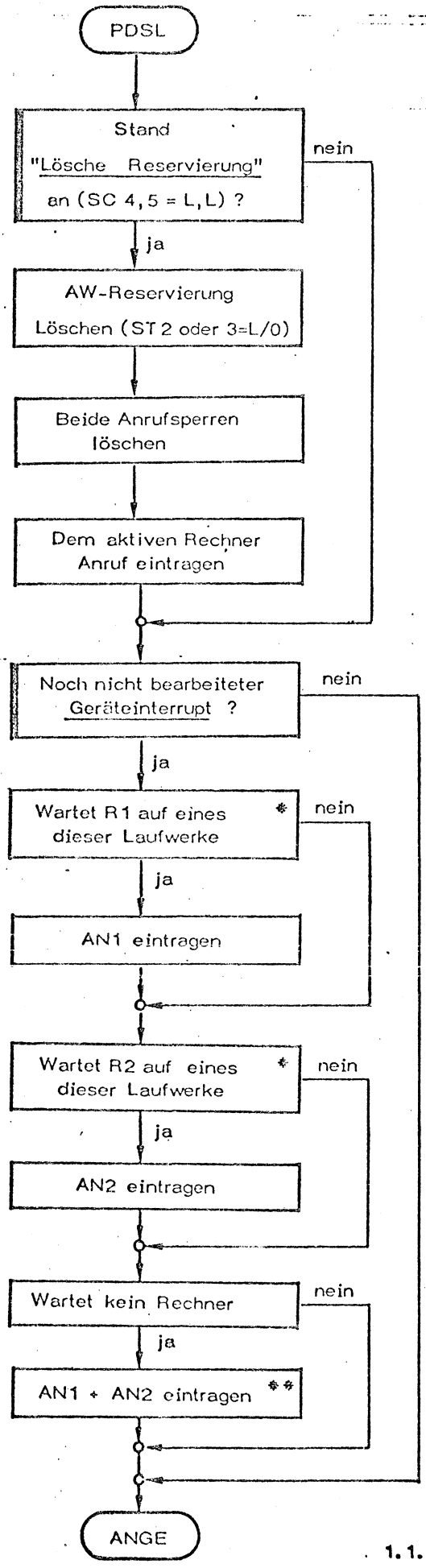
1.1. STARTPHASE (Bl.2)



1.1. STARTPHASE (Bl. 2a)



1.1. STARTPHASE (Bl. 2b)



Anrufbehandlung

Es gibt mehrere Gründe einen Anruf abzugeben:

1. Es stand der Befehl "Lösche Reservierung" an.
2. Es steht ein Geräteinterrupt von einem (oder mehreren) Laufwerk (en) an.
3. Ein Anruf ist im AW gespeichert, d.h. er konnte zuvor nicht abgegeben werden.
4. Das AW wurde freigegeben und war zuvor vorreserviert.

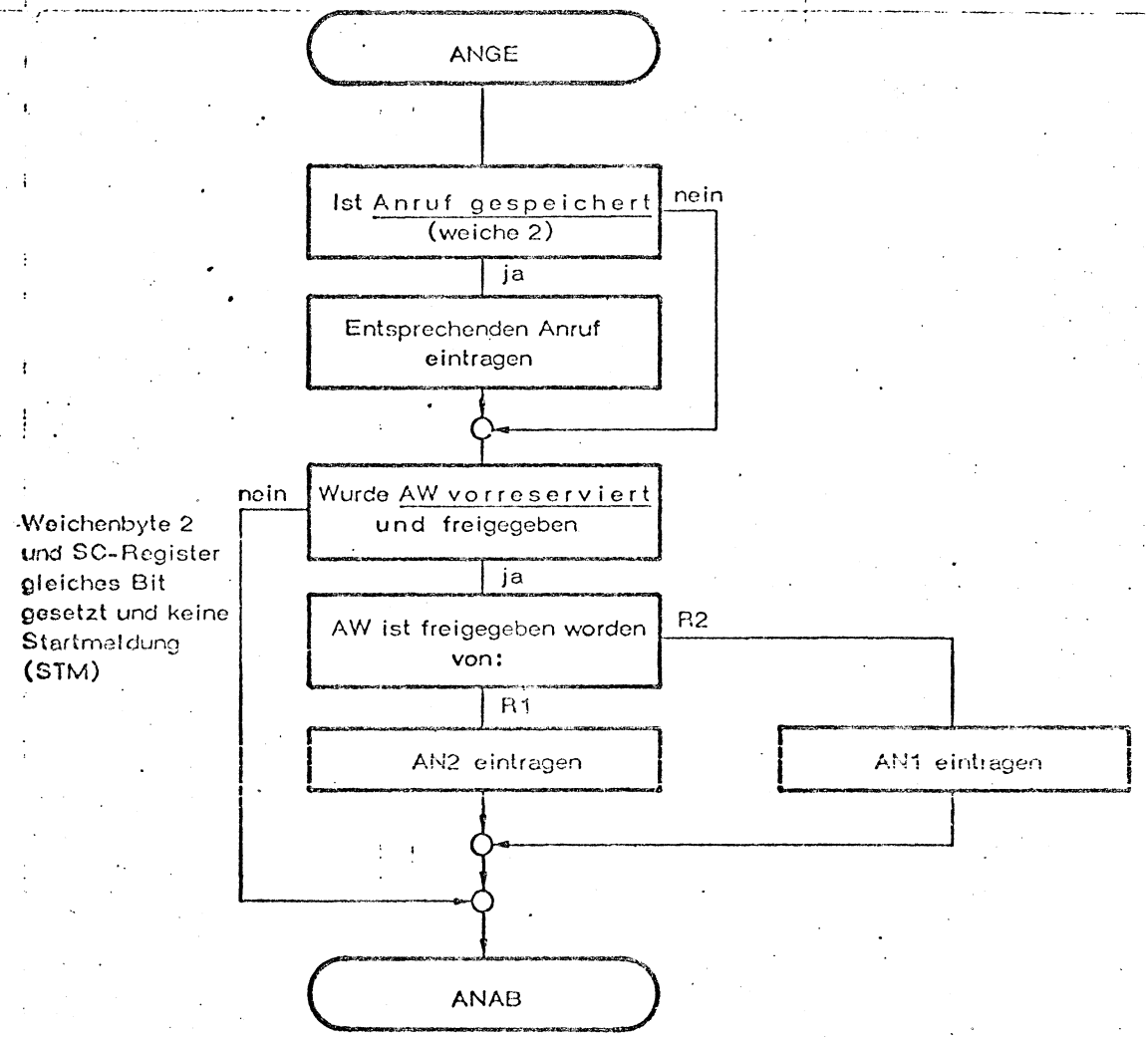
Die Anrufgründe werden der Reihe nach abgefragt und falls ein (oder mehrere) Anruf (e) gegeben werden soll (en), wird dieser in einem Anrufsammler (Register DR) eingetragen (aufgeodert). Danach wird geprüft, welcher Rechner einen Anruf erhalten darf und soll und dieser wird abgegeben. Anrufe dürfen gegeben werden, wenn:

1. Anruf gegeben werden soll und
2. keine Anrufsperre sitzt und
3. AW für Rechner reserviert oder frei ist.

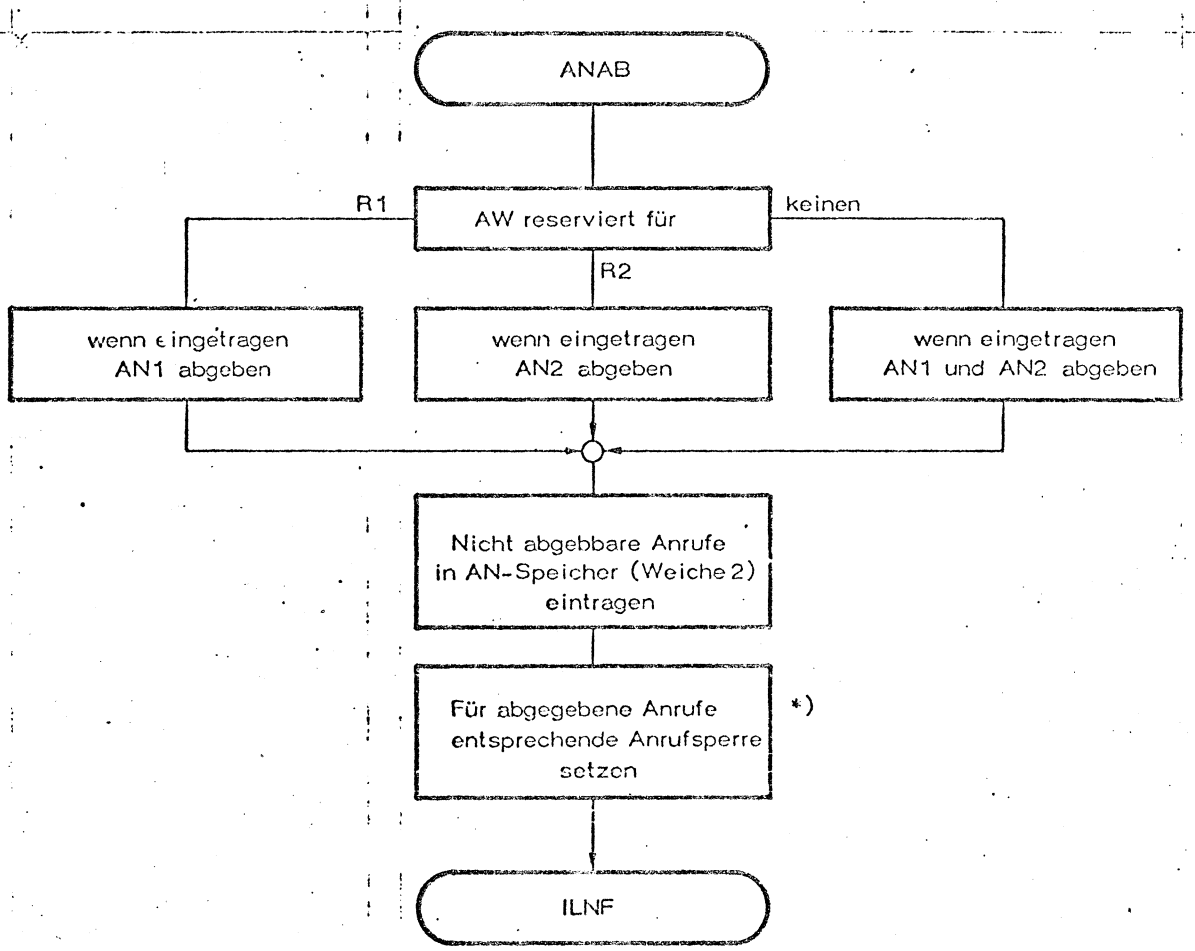
* d.h. der Rechner hat vorher einen Positionier-, Offset- oder Normierbefehl gegeben.

** Wenn kein Rechner auf Anruf wartet, so wird angenommen, daß es sich um einen Attentioninterrupt handelt.

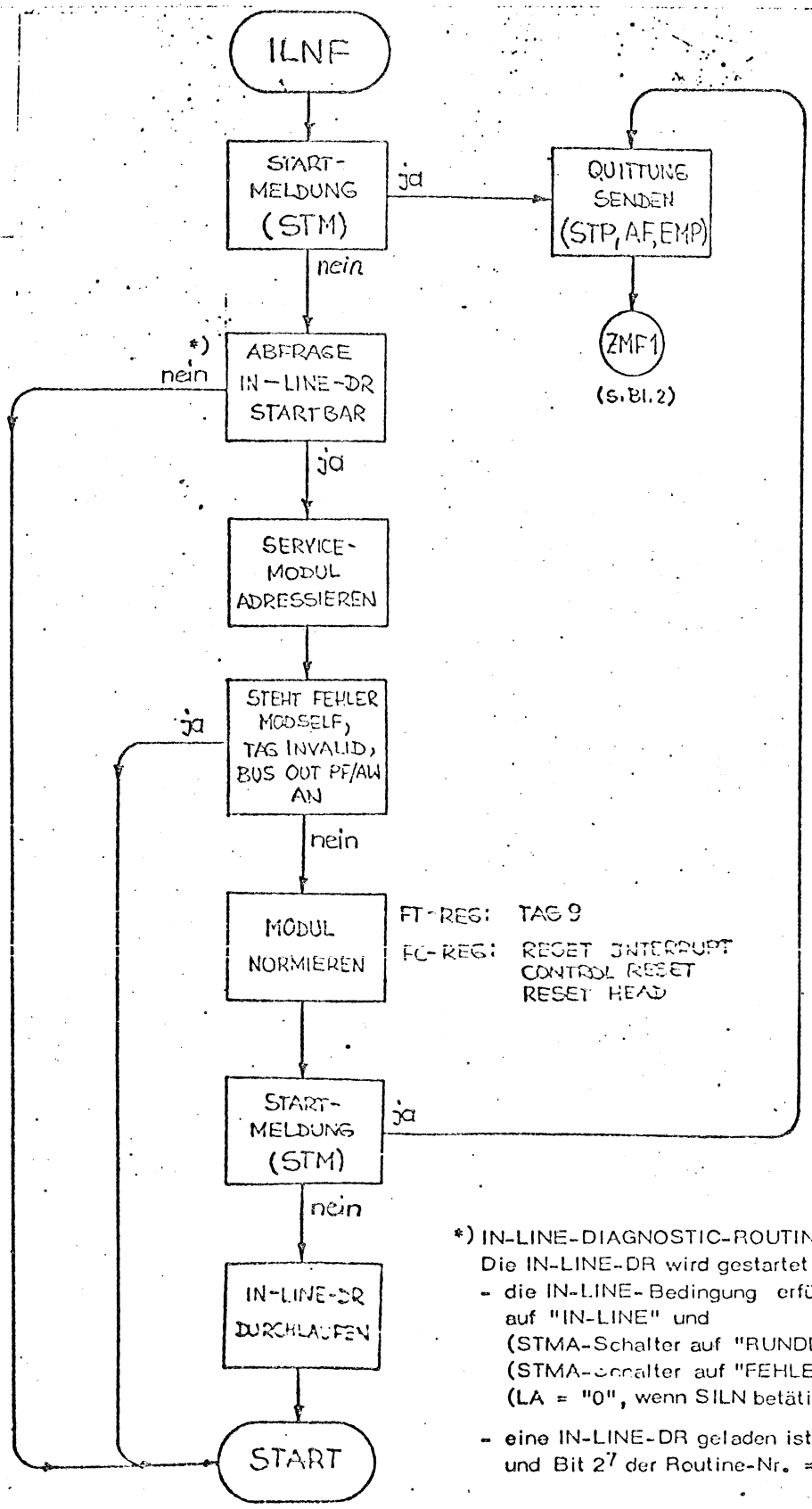
PE



1.1. STARTPHASE (Bl. 2d)

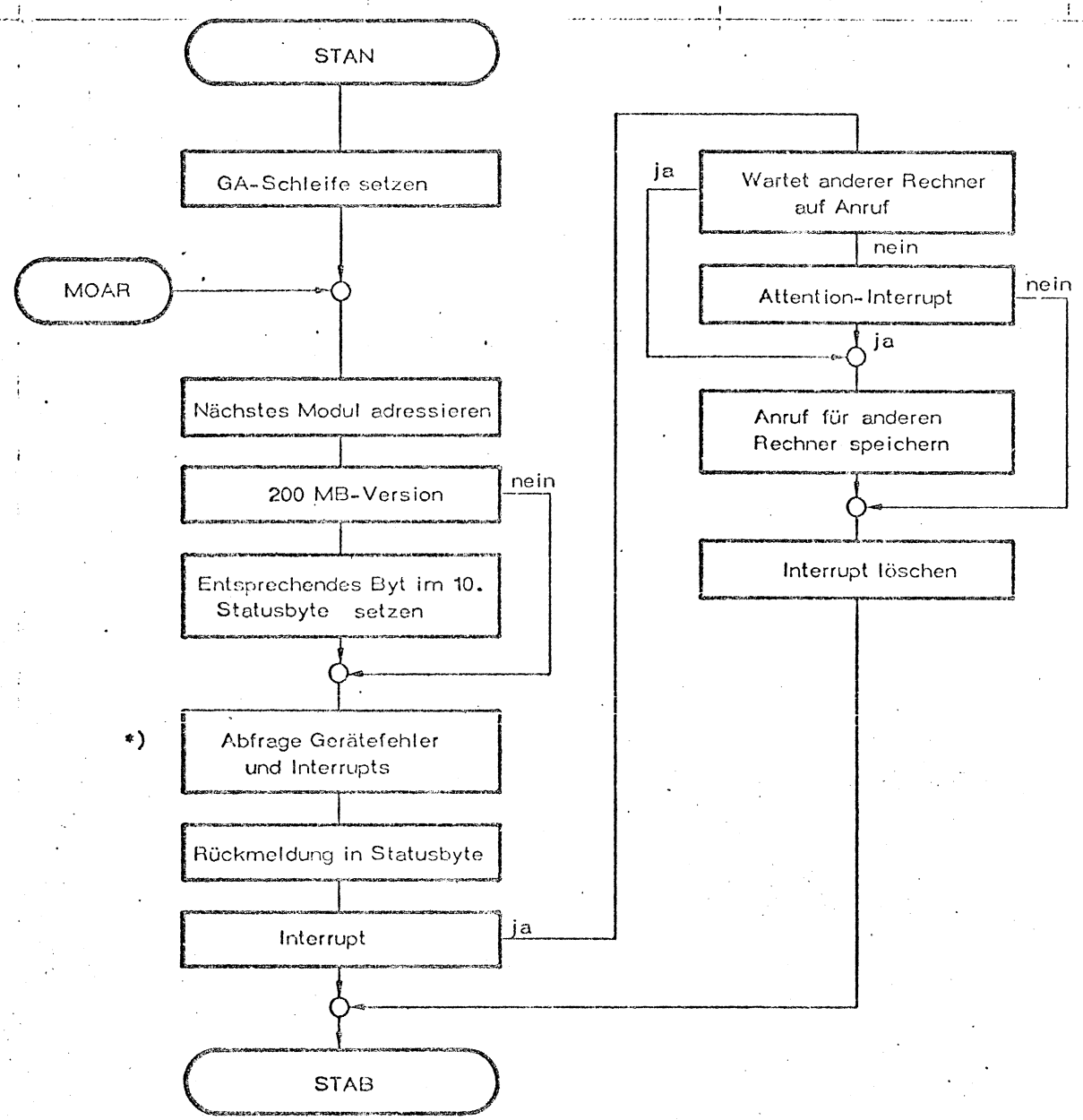


*) Die Anrufsperre wird gelöscht bei der "Statusabfrage nach Anruf". Daher muß jeder Anruf vom Rechner mit "Statusabfrage nach Anruf" beantwortet werden.



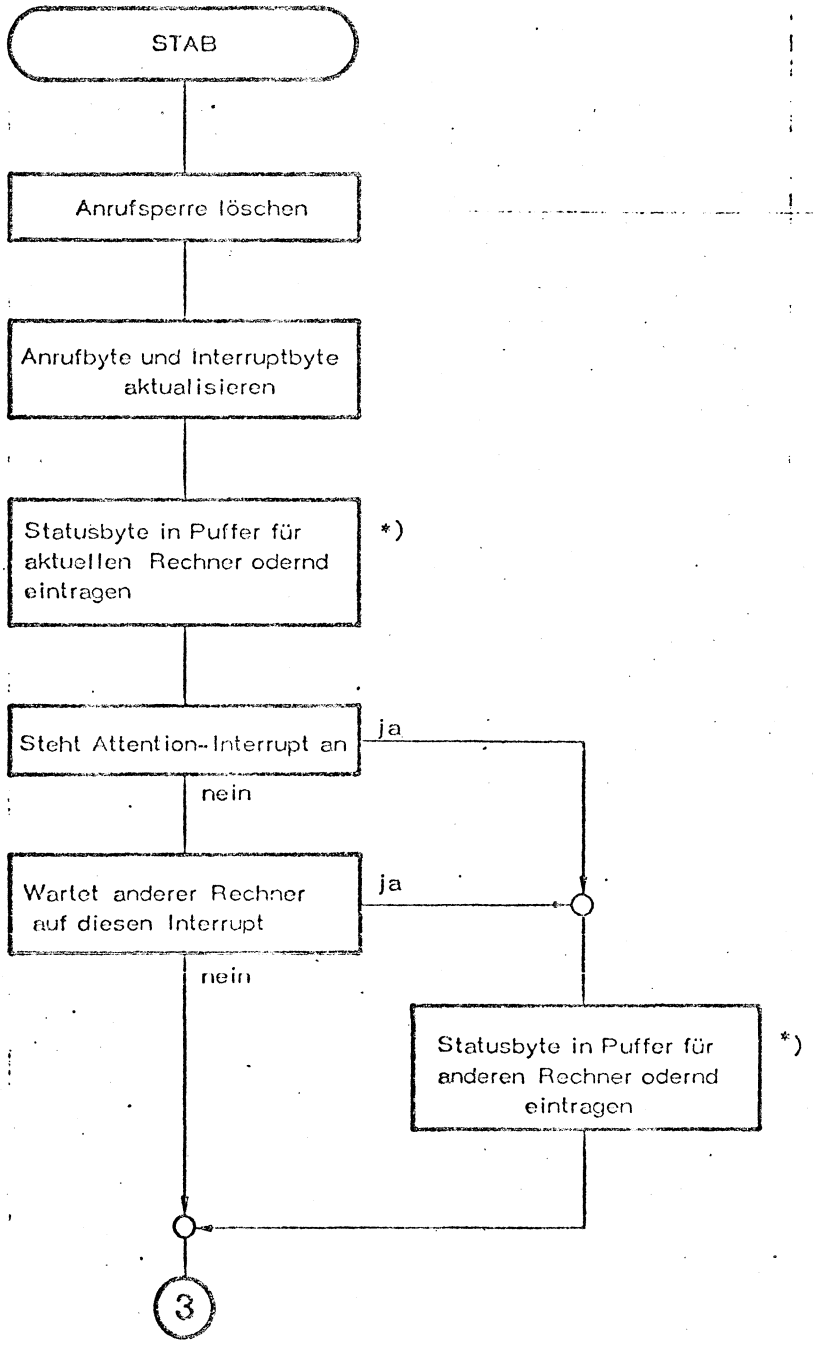
*) IN-LINE-DIAGNOSTIC-ROUTINE
Die IN-LINE-DR wird gestartet, wenn:
- die IN-LINE-Bedingung erfüllt ist, d.h. SB-Schalter auf "IN-LINE" und
(STMA-Schalter auf "RUNDLAUF" oder
(STMA-Schalter auf "FEHLER" und LA = "0"))
(LA = "0", wenn SILN betätigt wird), sowie
- eine IN-LINE-DR geladen ist, d.h. Routine-Nr. ≠ "0" und Bit 2^7 der Routine-Nr. = "0".

1.1 STARTPHASE (Bl. 3)



*) Gerätefehler kann evtl. über den Befehl "Modul normieren" gelöscht werden.

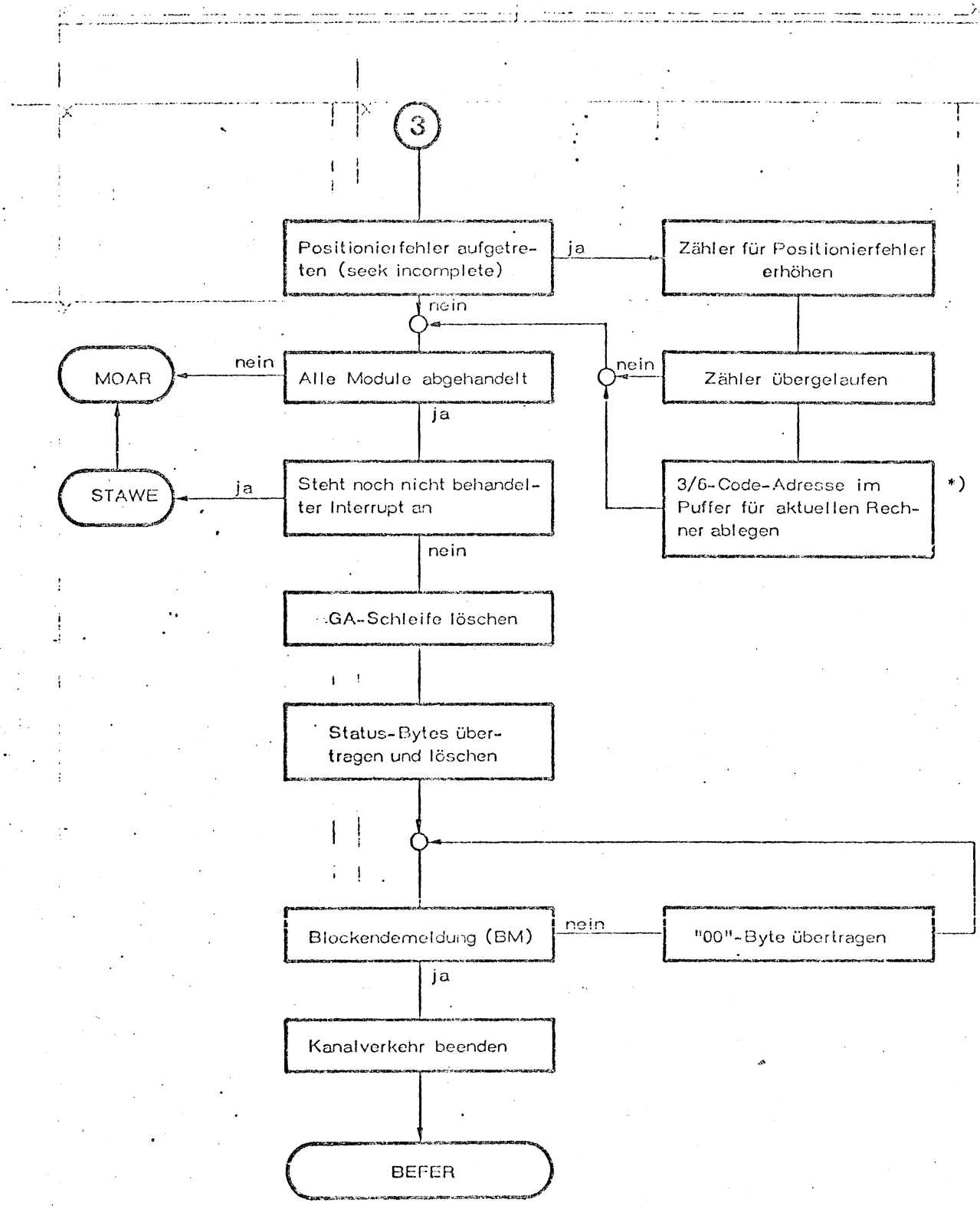
1.2. STATUSABFRAGE BEI ANRUF (Bl. 1)



*) Im DSP ist für beide Rechner je ein Puffer für die Statusbytes.
Die Statusbytes werden stets in den Puffer des mit dem AW verkehrenden Rechners geschrieben.
In den anderen Puffer wird nur dann das aktuelle Statusbyte geschrieben, wenn

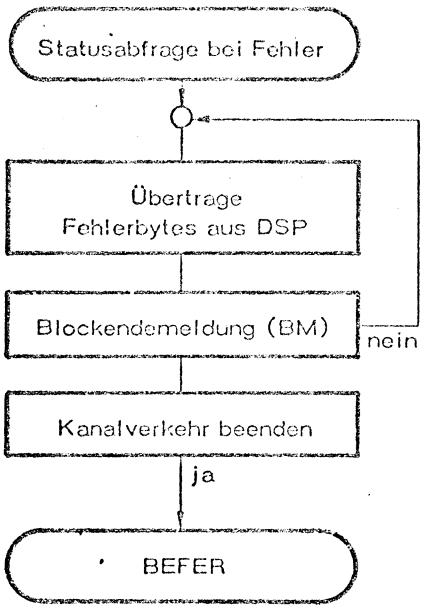
- Bit 2^2 (Attention) ansteht oder
- ein Modul "Interrupt" meldet und der andere Rechner auf Anruf dieses Moduls wartet.

1.2. STATUSABFRAGE BEI ANRUF (Bl. 1a)

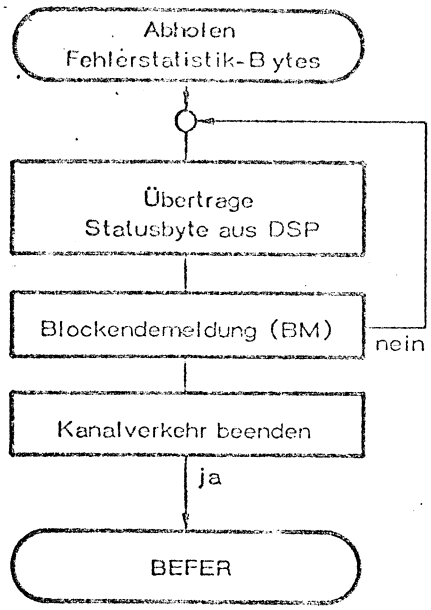


*) Dieses Byte wird als 9. Statusbyte übertragen

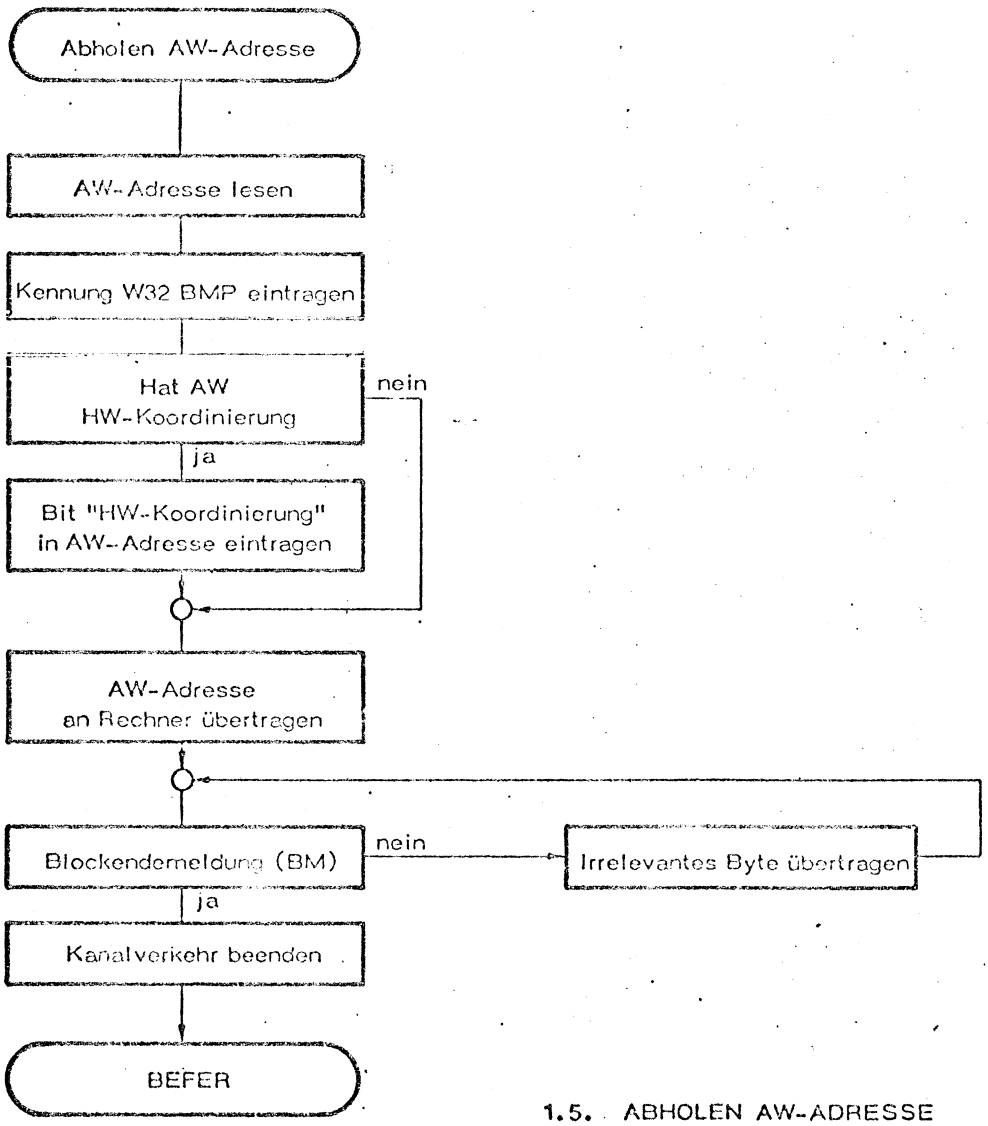
1.2. STATUSABFRAGE BEI ANRUF (Bl. 1b)



1.3. STATUSABFRAGE BEI FEHLER

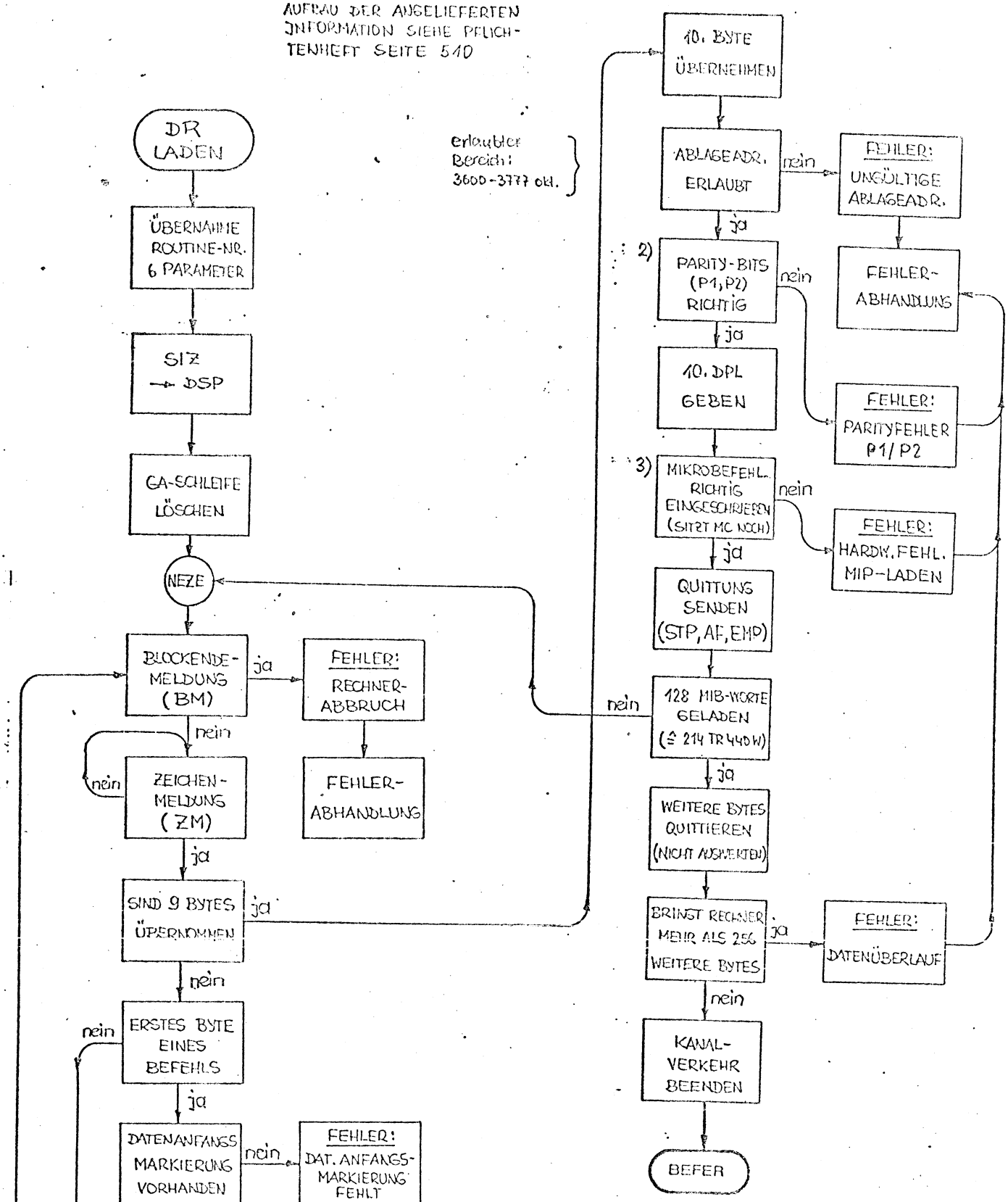


1.4. ABHOLEN FEHLERSTATISTIK-BYTES



1.5. ABHOLEN AW-ADRESSE PE

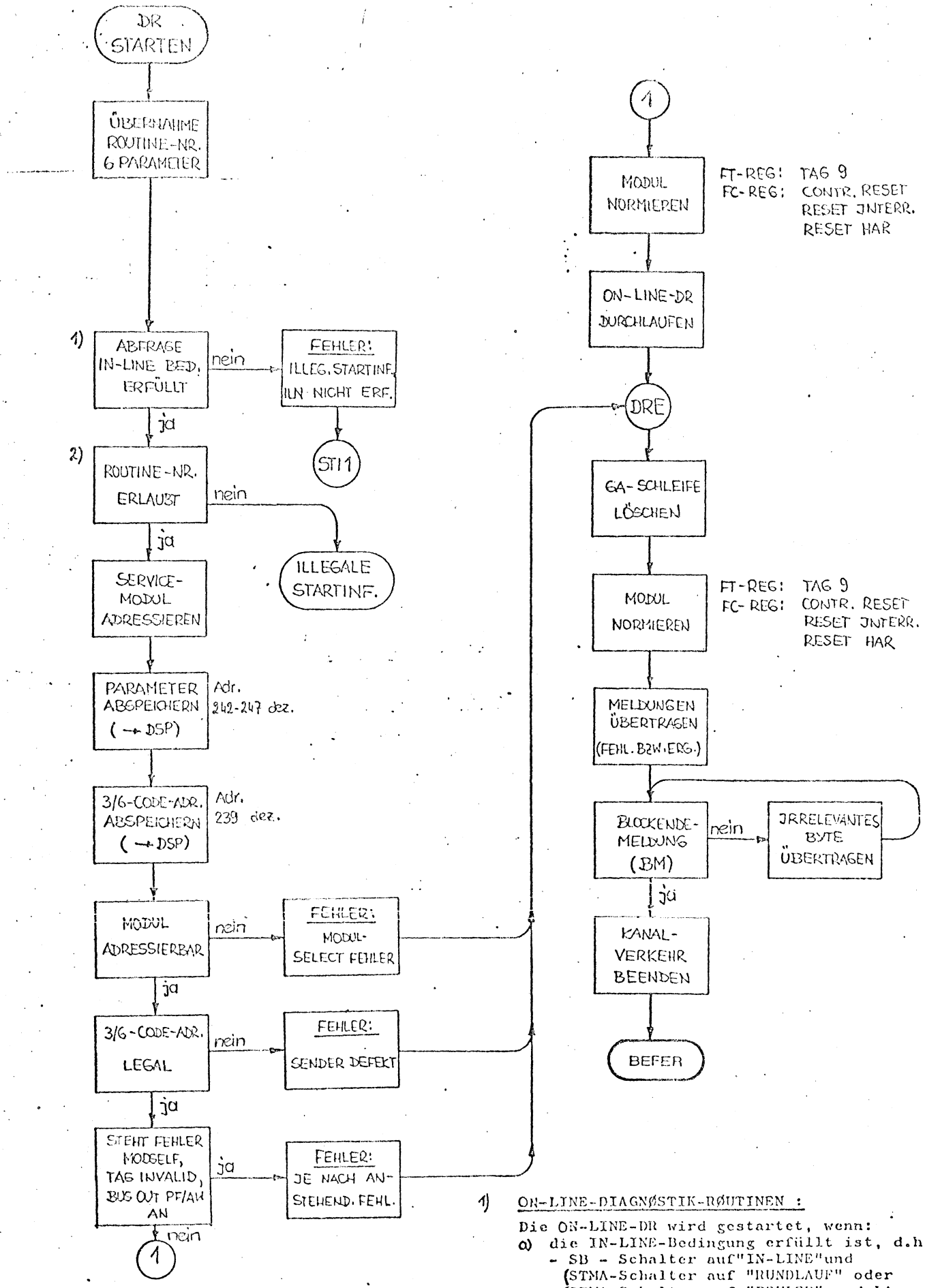
AUFBAU DER ANGELIEFERTEN INFORMATION SIEHE PFLICHTENHEFT SEITE 510



1) Mit STP erfolgt Durchschaltung des Bytes vom Kanal → KA-Register.

2) Das Parity Bit der Adresse (PA) überprüft die Hardware.

3) Nach dem 10. DPL lädt Hardware den MIKRO-BEFEHL in den MIP-Speicher. Die Taktkette wird angehalten.
Tritt ein Fehler auf (PA falsch, Vergleich beim Wiederauslesen falsch) löscht Hardware den MC-Merker.
Nach dem 10. DPL ist ein Leerbefehl nötig; BSH und BSN wird aber ausgewertet.



2) Die Routine-Nr. muß bei „LADEN" und „STARTEN" gleich sein.

1) ON-LINE-DIAGNØSTIK-RØUTINEN :

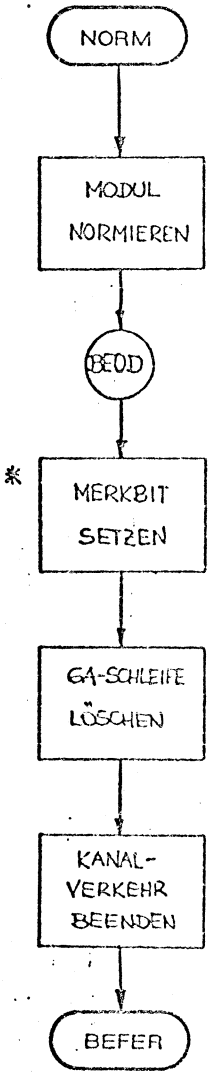
Die ON-LINE-DR wird gestartet, wenn:

a) die IN-LINE-Bedingung erfüllt ist, d.h.
   - SB - Schalter auf "IN-LINE" und
   (STMA-Schalter auf "RUNDLAUF" oder
   (STMA-Schalter auf "FEHLER" und LA = 0))

   (LA=0, wenn SILN betätigt wird)

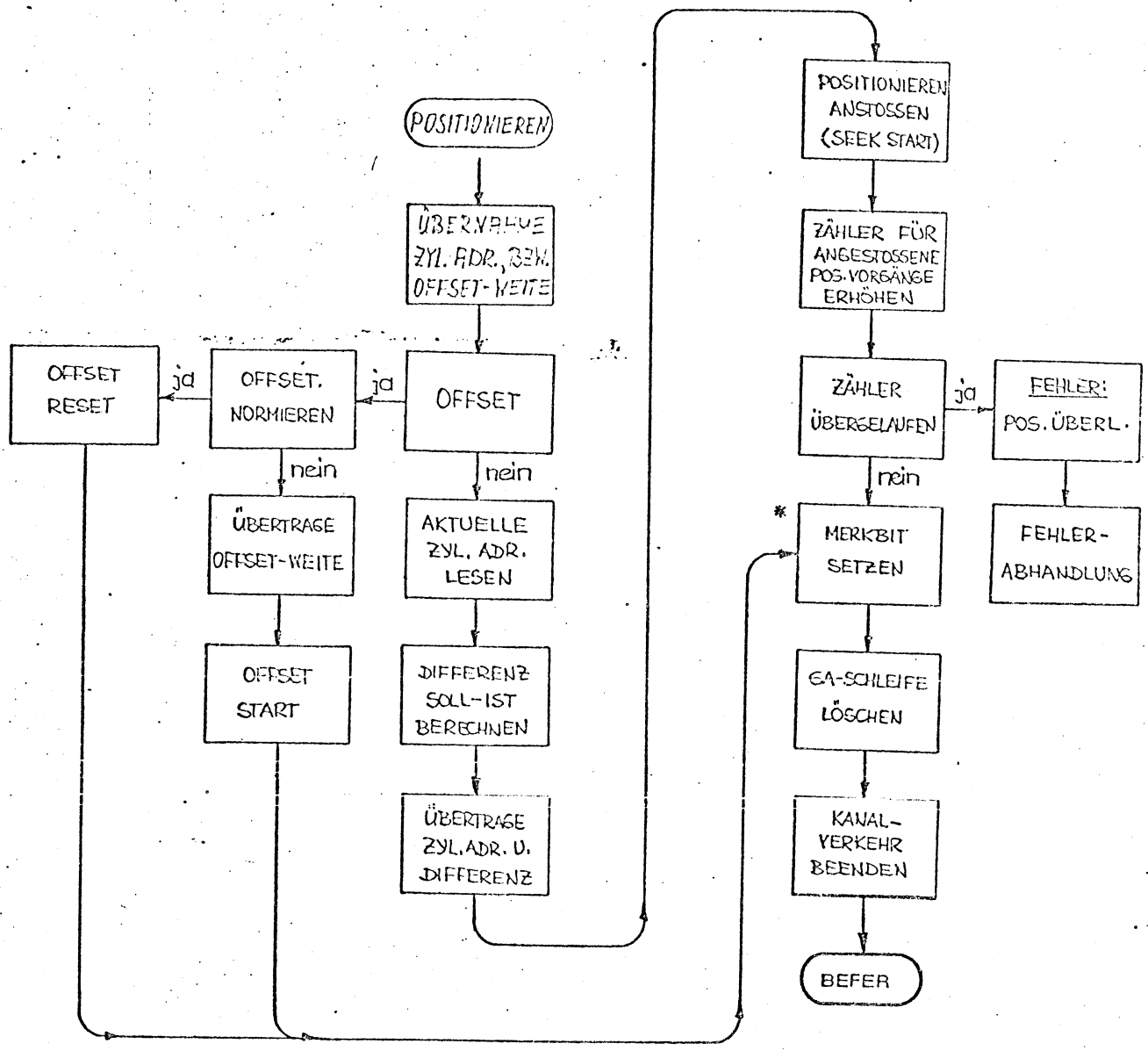
b) eine ON-LINE-DR geladen ist, d.h.
   - Routine-Nr. ≠ 0 und Bit 2^7 der Routine-Nr. = „L".

1.7. DIAGNOSTIKPROGRAMM STARTEN



* Hierfür werden im DSP zwei Bytes geführt (R1 : 37, R2 : 38), in denen modulspezifisch vermerkt ist, welcher Rechner auf den Anruf welchen Moduls wartet.

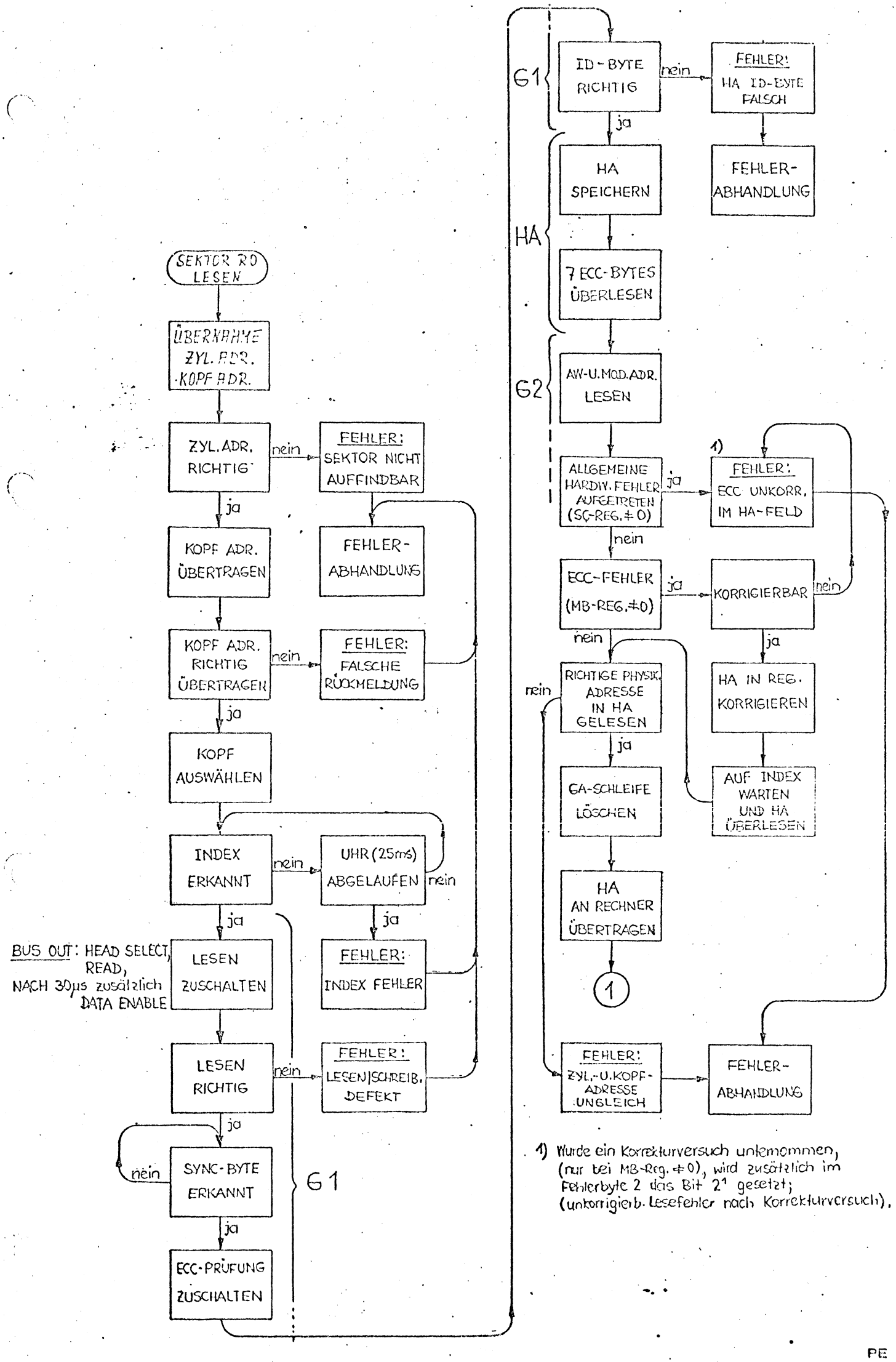
1.8. MODUL NORMIEREN



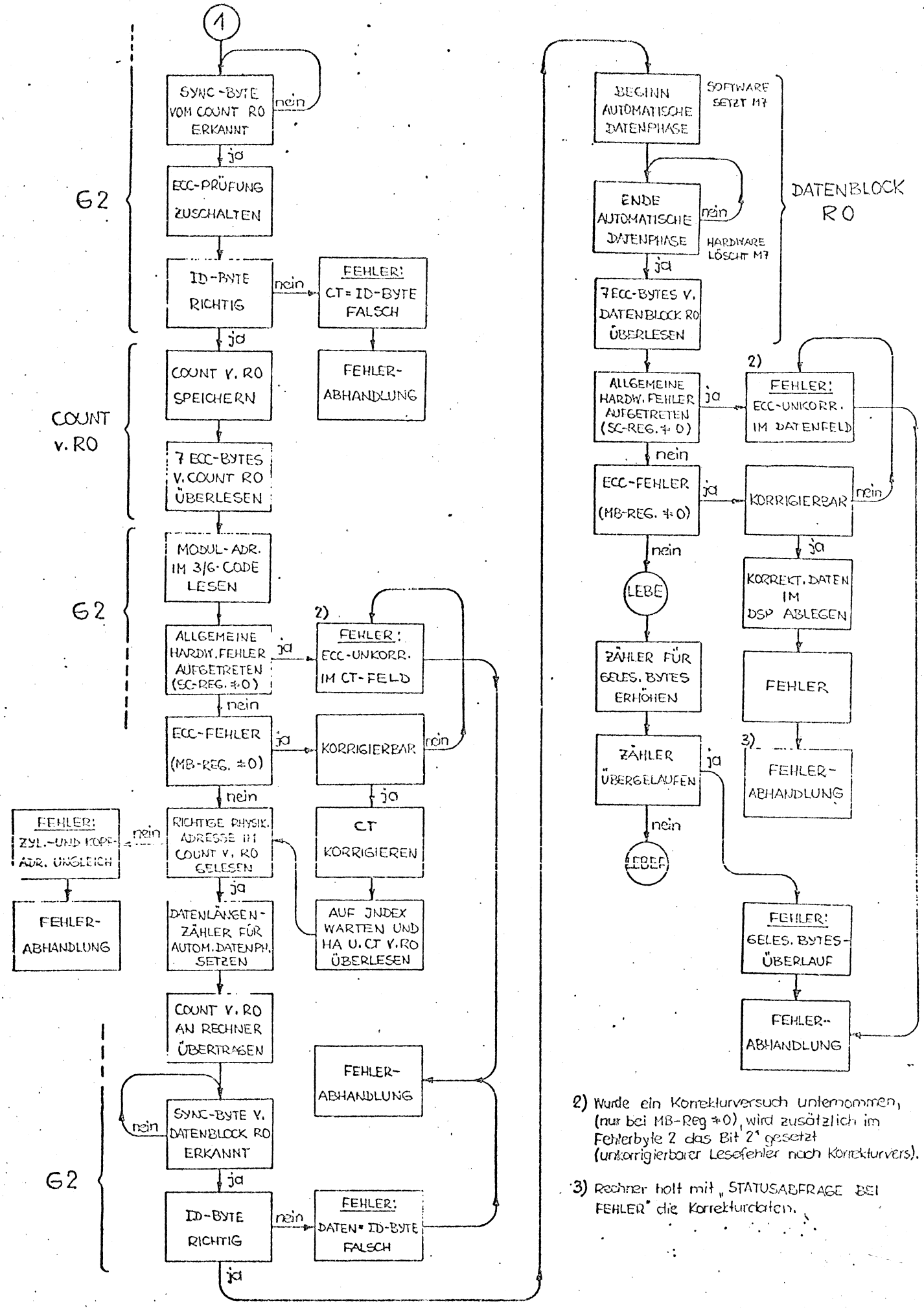
* Merkbit setzen für: Dieser Rechner wartet auf Anruf dieses Moduls.

Hierfür werden im DSP zwei Bytes geführt (R1: 37, R2: 38), in deren modulspezifisch vermerkt ist, welcher Rechner auf den Anruf welchen Moduls wartet.

1.9. POSITIONIEREN


SEKTOR RD LESEN
ÜBERNAHME ZYL. ADR. KOPF ADR.
ZYL. ADR. RICHTIG
nein
FEHLER: SEKTOR NICHT AUFFINDBAR
ja
KOPF ADR. ÜBERTRAGEN
FEHLER-ABHANDLUNG
KOPF ADR. RICHTIG ÜBERTRAGEN
nein
FEHLER: FALSCHE RÜCKMELDUNG
ja
KOPF AUSWÄHLEN
INDEX ERKANNT
nein
UHR (25ms) ABGELAUFEN
nein
ja
ja
BUS OUT: HEAD SELECT, READ, NACH 30µs zusätzlich DATA ENABLE
LESEN ZUSCHALTEN
FEHLER: INDEX FEHLER
LESEN RICHTIG
nein
FEHLER: LESEN/SCHREIB. DEFEKT
ja
nein
SYNC-BYTE ERKANNT
ja
ECC-PRÜFUNG ZUSCHALTEN
G1
ID-BYTE RICHTIG
nein
FEHLER: HA ID-BYTE FALSCH
ja
HA SPEICHERN
FEHLER-ABHANDLUNG
HA
7 ECC-BYTES ÜBERLESEN
G2
AW-U. MOD. ADR. LESEN
ALLGEMEINE HARDW. FEHLER AUFGETRETEN (SC-REG. ≠ 0)
ja
1)
FEHLER: ECC UNKORR. IM HA-FELD
nein
ECC-FEHLER (MB-REG. ≠ 0)
ja
KORRIGIERBAR
nein
ja
nein
RICHTIGE PHYSIK. ADRESSE IN HA GELESEN
nein
HA IN REG. KORRIGIEREN
ja
GA-SCHLEIFE LÖSCHEN
AUF INDEX WARTEN UND HA ÜBERLESEN
HA AN RECHNER ÜBERTRAGEN
1
FEHLER: ZYL.- U. KOPF-ADRESSE UNGLEICH
FEHLER-ABHANDLUNG
1) Wurde ein Korrekturversuch unternommen, (nur bei MB-Reg. ≠ 0), wird zusätzlich im Fehlerbyte 2 das Bit 2^1 gesetzt; (unkorrigierb. Lesefehler nach Korrekturversuch).

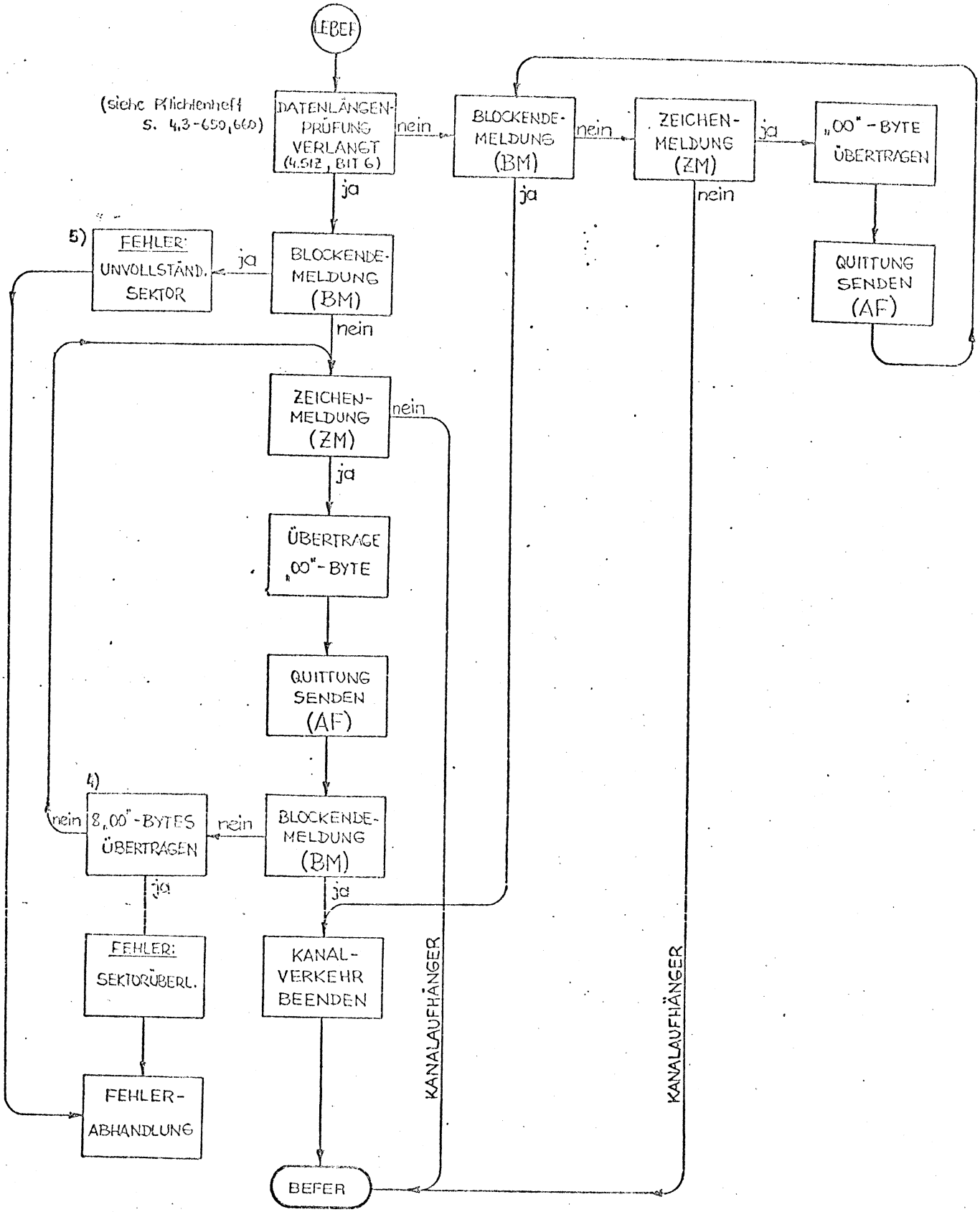
PE



2) Wurde ein Korrekturversuch unternommen, (nur bei MB-Reg ≠ 0), wird zusätzlich im Fehlerbyte 2 das Bit 2⁴ gesetzt (unkorrigierbarer Lesefehler nach Korrekturvers).

3) Rechner holt mit „STATUSABFRAGE BEI FEHLER" die Korrekturdaten.

1.10. SEKTOR RO LESEN (BL. 2)

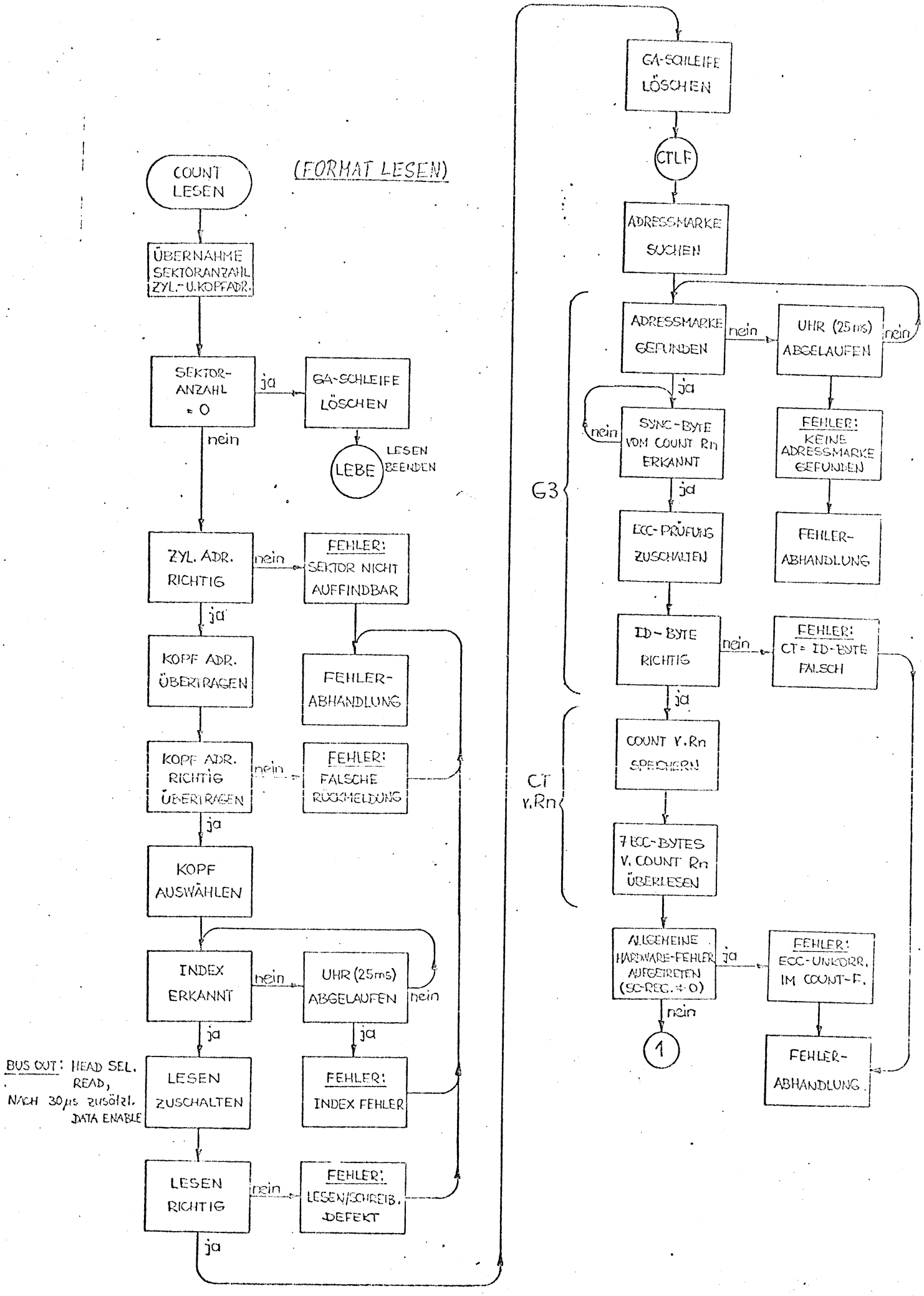


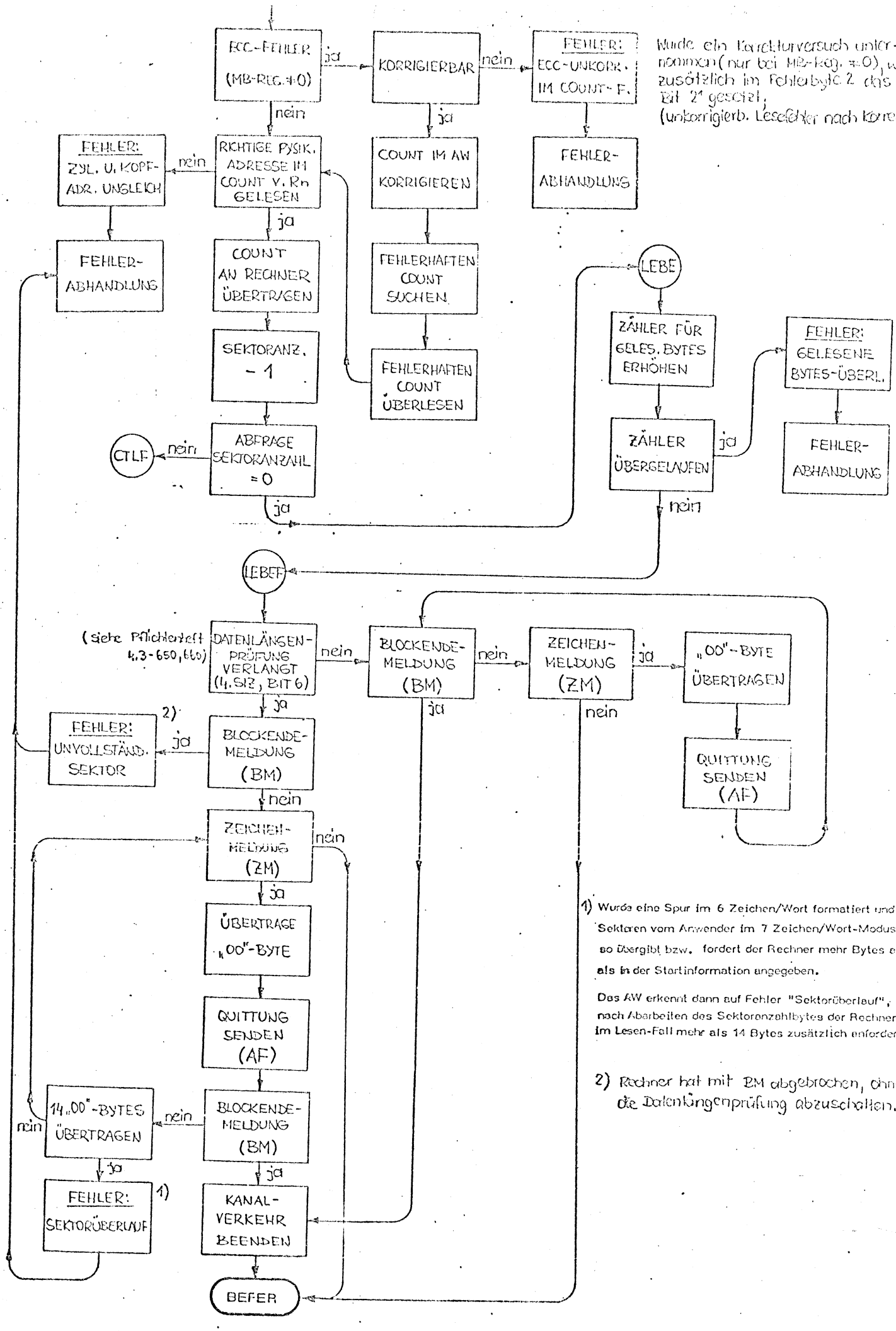
4) Fordert der Rechner mehr Datenbytes an als in der Startinformation angegeben, so sendet das AW „00"-Bytes zum Rechner, bis dieser die Übertragung abbricht.

Fordert der Rechner mehr als 8 „00"-Bytes an, so bricht das AW mit Fehlereingriff ab.

5) Rechner hat mit BM abgebrochen, ohne die Datenlängenprüfung abzuschalten.

1.10. SEKTOR RO LESEN (BL.3)


COUNT LESEN
(FORMAT LESEN)
ÜBERNAHME SEKTORANZAHL ZYL.- U. KOPFADR.
SEKTOR-ANZAHL = 0
ja
GA-SCHLEIFE LÖSCHEN
nein
LEBE
LESEN BEENDEN
ZYL. ADR. RICHTIG
nein
FEHLER: SEKTOR NICHT AUFFINDBAR
ja
KOPF ADR. ÜBERTRAGEN
FEHLER-ABHANDLUNG
KOPF ADR. RICHTIG ÜBERTRAGEN
nein
FEHLER: FALSCHE RÜCKMELDUNG
ja
KOPF AUSWÄHLEN
INDEX ERKANNT
nein
UHR (25ms) ABGELAUFEN
nein
ja
ja
BUS OUT: HEAD SEL. READ, NACH 30µs zusätzl. DATA ENABLE
LESEN ZUSCHALTEN
FEHLER: INDEX FEHLER
LESEN RICHTIG
nein
FEHLER: LESEN/SCHREIB. DEFEKT
ja
GA-SCHLEIFE LÖSCHEN
CTLF
ADRESSMARKE SUCHEN
ADRESSMARKE GEFUNDEN
nein
UHR (25ms) ABGELAUFEN
nein
ja
nein
SYNC-BYTE VOM COUNT Rn ERKANNT
FEHLER: KEINE ADRESSMARKE GEFUNDEN
G3
ja
ECC-PRÜFUNG ZUSCHALTEN
FEHLER-ABHANDLUNG
ID-BYTE RICHTIG
nein
FEHLER: CT = ID-BYTE FALSCH
ja
CT v. Rn
COUNT v. Rn SPEICHERN
7 ECC-BYTES v. COUNT Rn ÜBERLESEN
ALLGEMEINE HARDWARE-FEHLER AUFGETRETEN (SC-REG. ≠ 0)
ja
FEHLER: ECC-UNKORR. IM COUNT-F.
nein
1
FEHLER-ABHANDLUNG

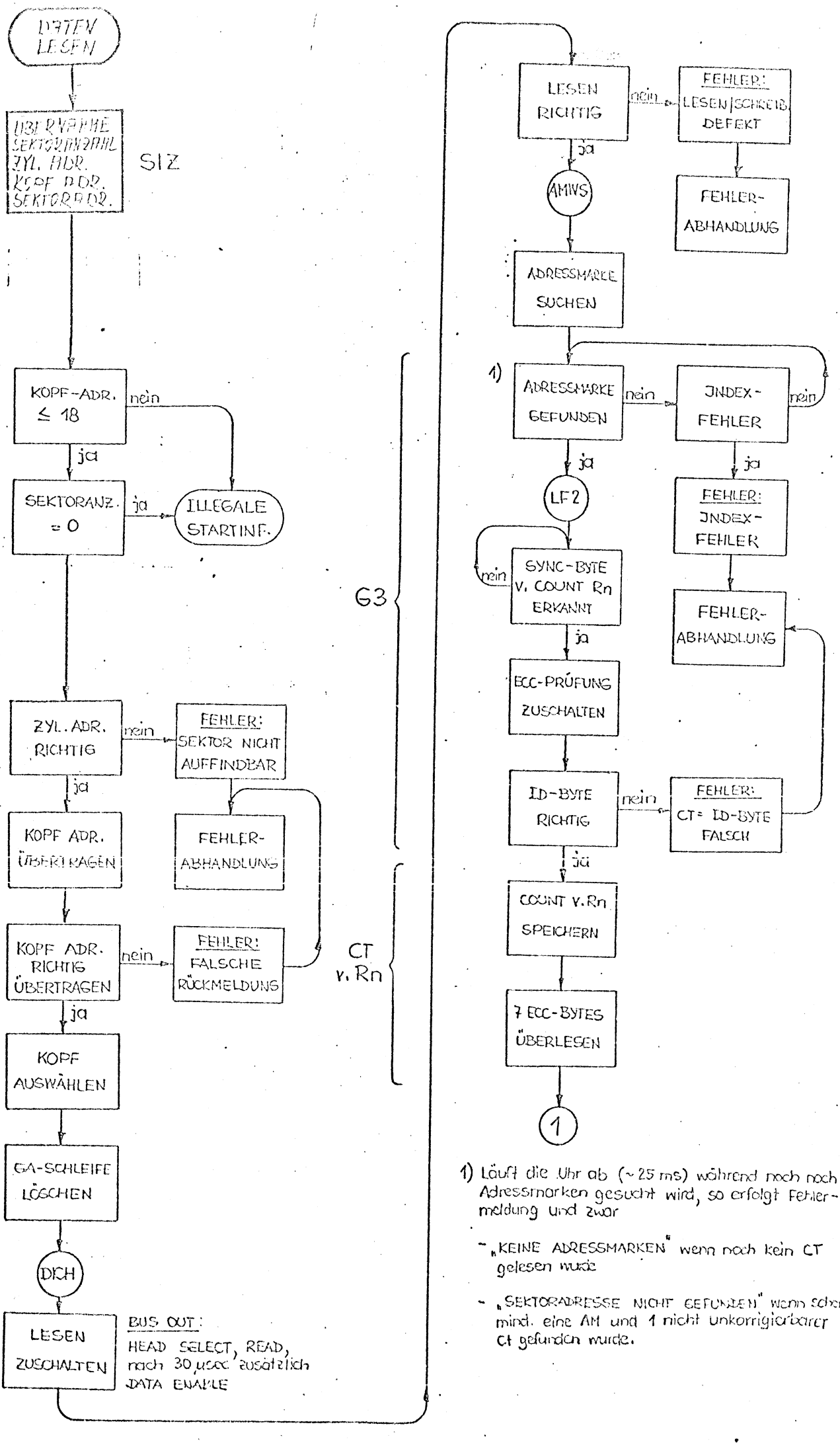


1) Wurde eine Spur im 6 Zeichen/Wort formatiert und die Sektoren vom Anwender im 7 Zeichen/Wort-Modus benutzt so übergibt bzw. fordert der Rechner mehr Bytes an als in der Startinformation angegeben.

Das AW erkennt dann auf Fehler "Sektorüberlauf", wenn nach Abarbeiten des Sektoranzahlbytes der Rechner im Lesen-Fall mehr als 14 Bytes zusätzlich anfordert.

2) Rechner hat mit BM abgebrochen, ohne die Datenlängenprüfung abzuschalten.

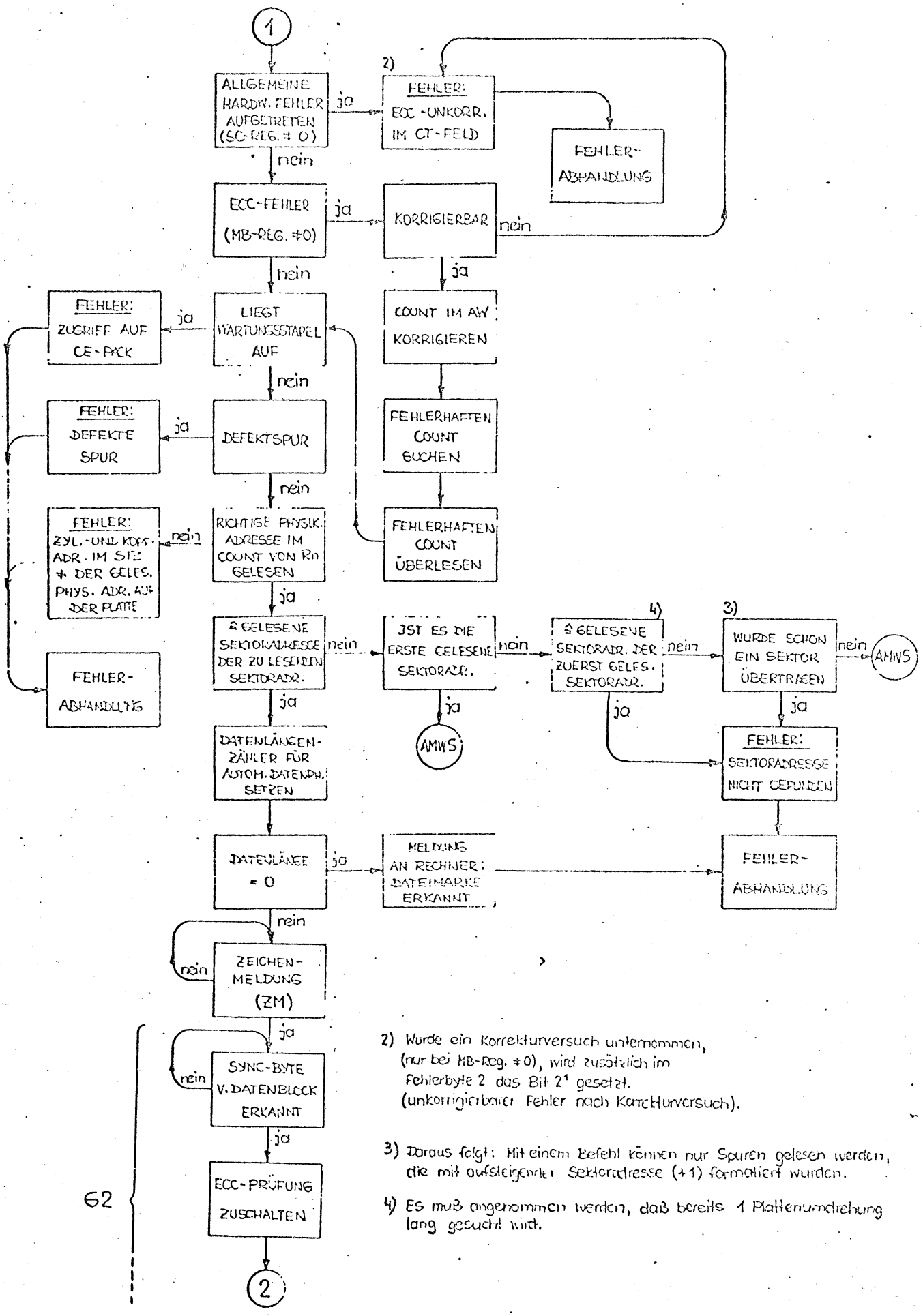
1.11. COUNT LESEN (BL.2)



1) Läuft die Uhr ab (~ 25 ms) während noch nach Adressmarken gesucht wird, so erfolgt Fehlermeldung und zwar

- „KEINE ADRESSMARKEN" wenn noch kein CT gelesen wurde
- „SEKTORADRESSE NICHT GEFUNDEN" wenn schon mind. eine AM und 1 nicht unkorrigierbarer CT gefunden wurde.

PE



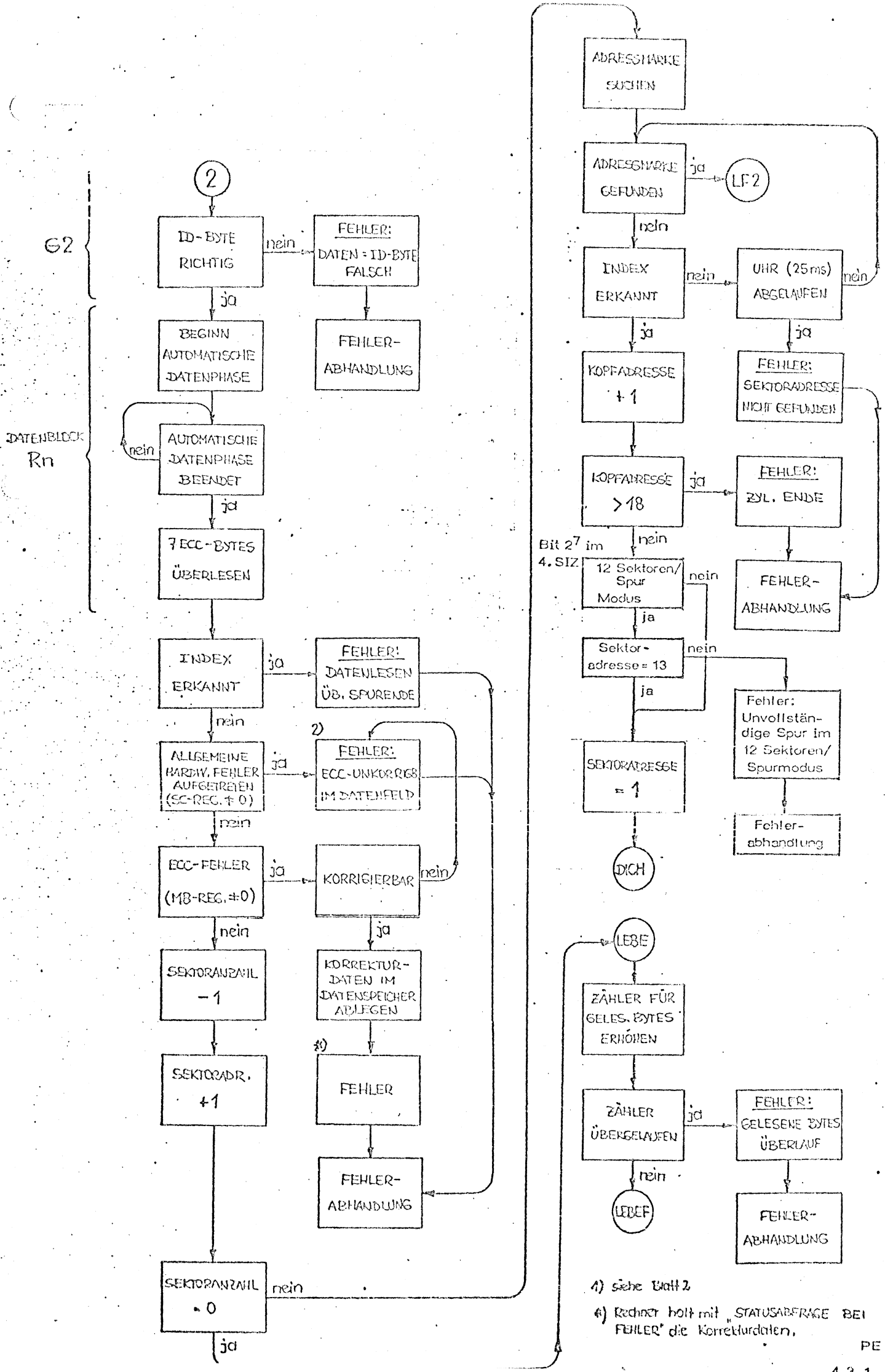
62

2) Wurde ein Korrekturversuch unternommen, (nur bei MB-Reg. ≠0), wird zusätzlich im Fehlerbyte 2 das Bit 2^1 gesetzt. (unkorrigierbarer Fehler nach Korrekturversuch).

3) Daraus folgt: Mit einem Befehl können nur Spuren gelesen werden, die mit aufsteigender Sektoradresse (+1) formatiert wurden.

4) Es muß angenommen werden, daß bereits 1 Plattenumdrehung lang gesucht wird.

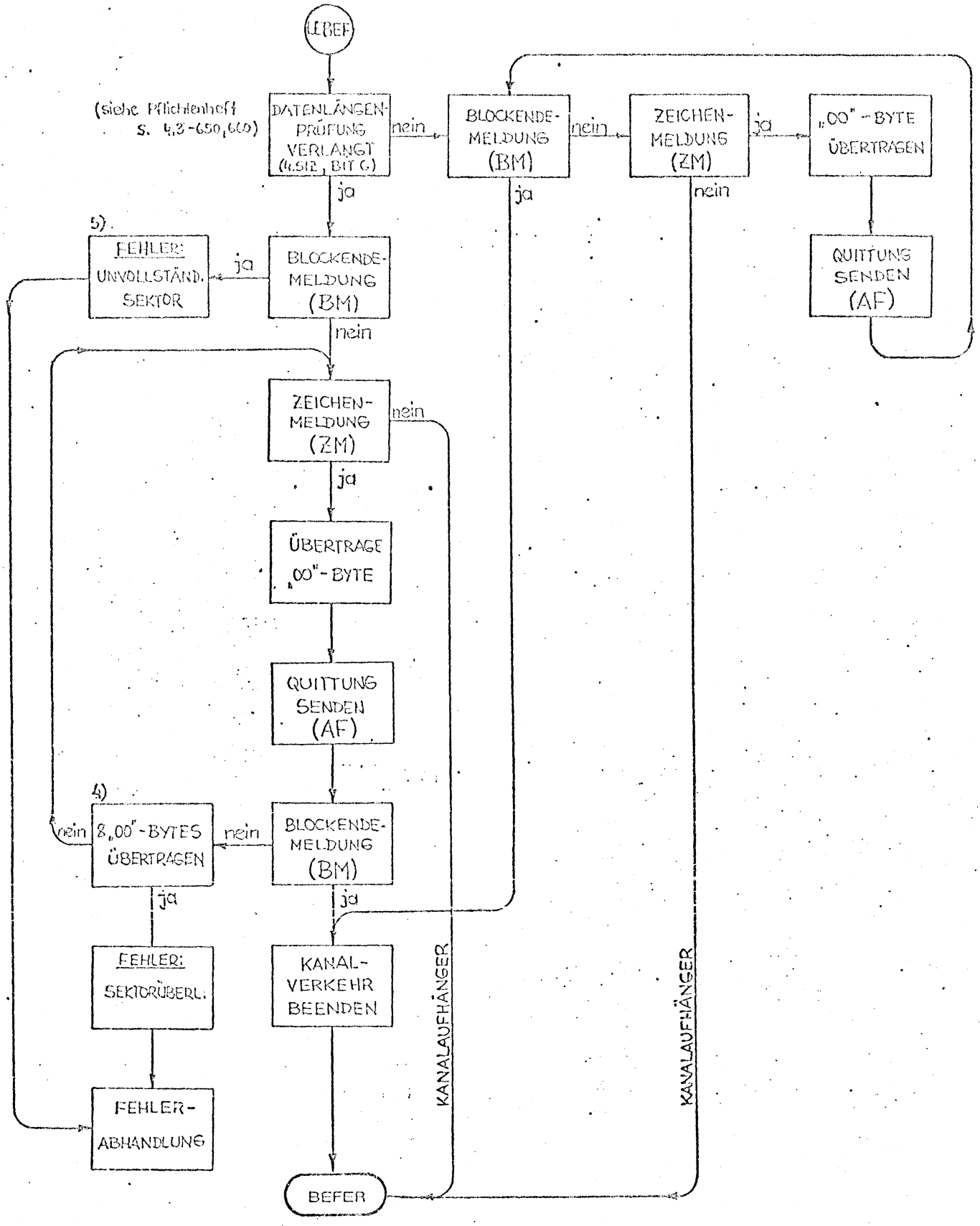
1.12. DATEN LESEN (BL 2)



1) siehe Blatt 2

4) Rechner holt mit „STATUSABFRAGE BEI FEHLER" die Korrekturdaten.

PE

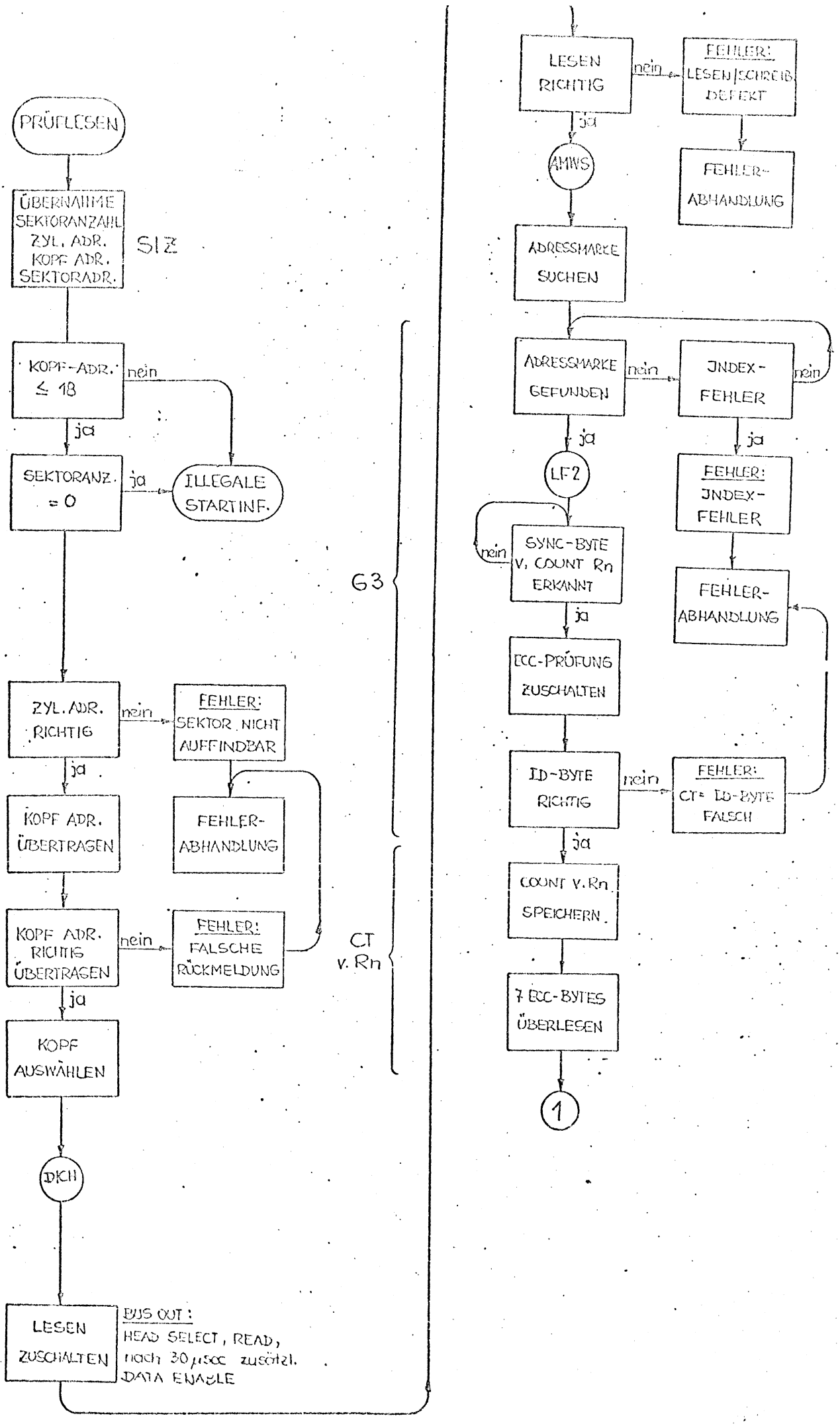


4) Fordert der Rechner mehr Datenbytes an als in der Startinformation angegeben, so sendet das AW „00"-Bytes zum Rechner, bis dieser die Übertragung abbricht.

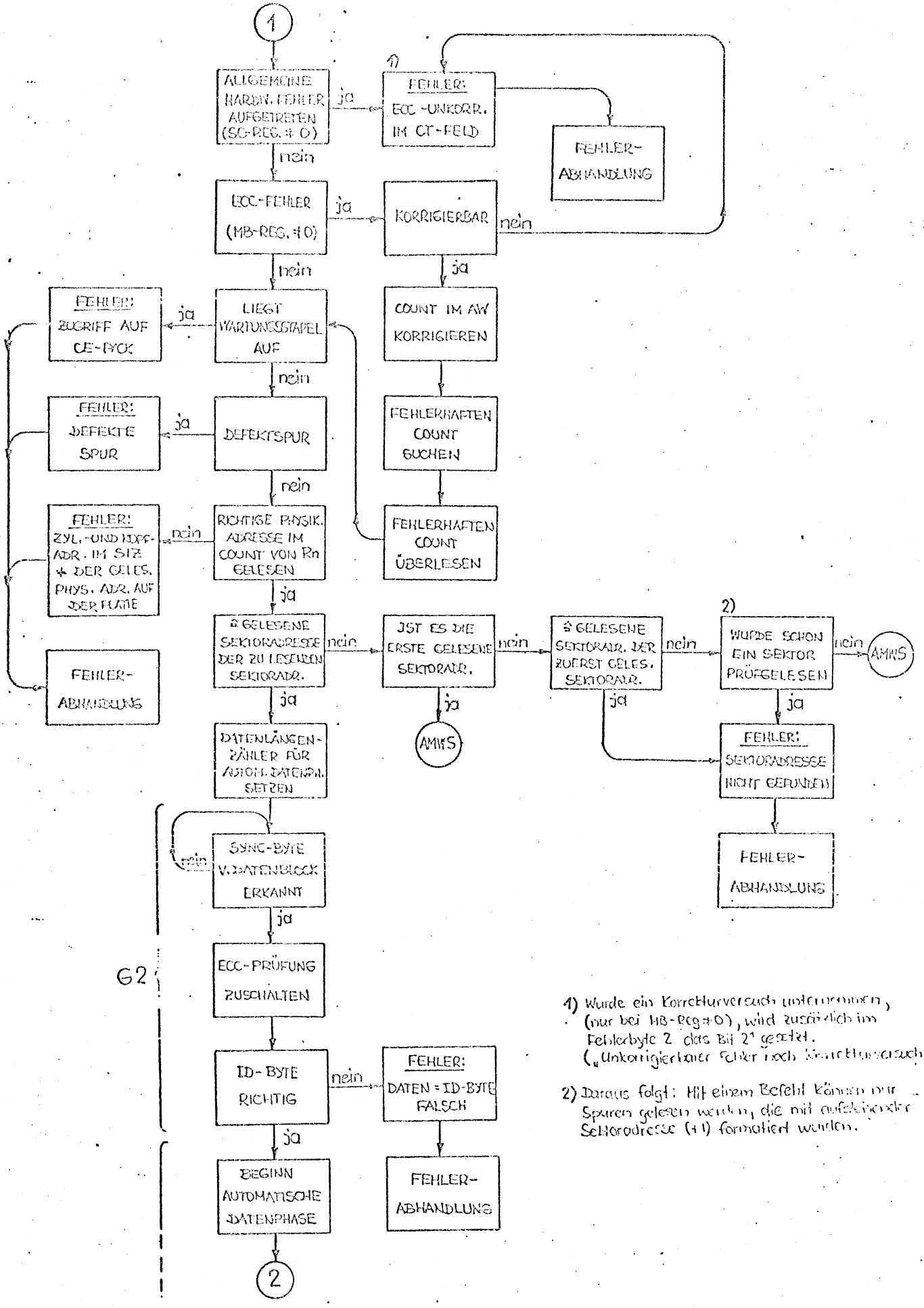
Fordert der Rechner mehr als 8 „00"-Bytes an, so bricht das AW mit Fehlereingriff ab.

5) Rechner hat mit BM abgebrochen (Teilblock lesen), ohne die Datenlängenprüfung abzuschalten.

1.12. DATEN LESEN (BL.4)



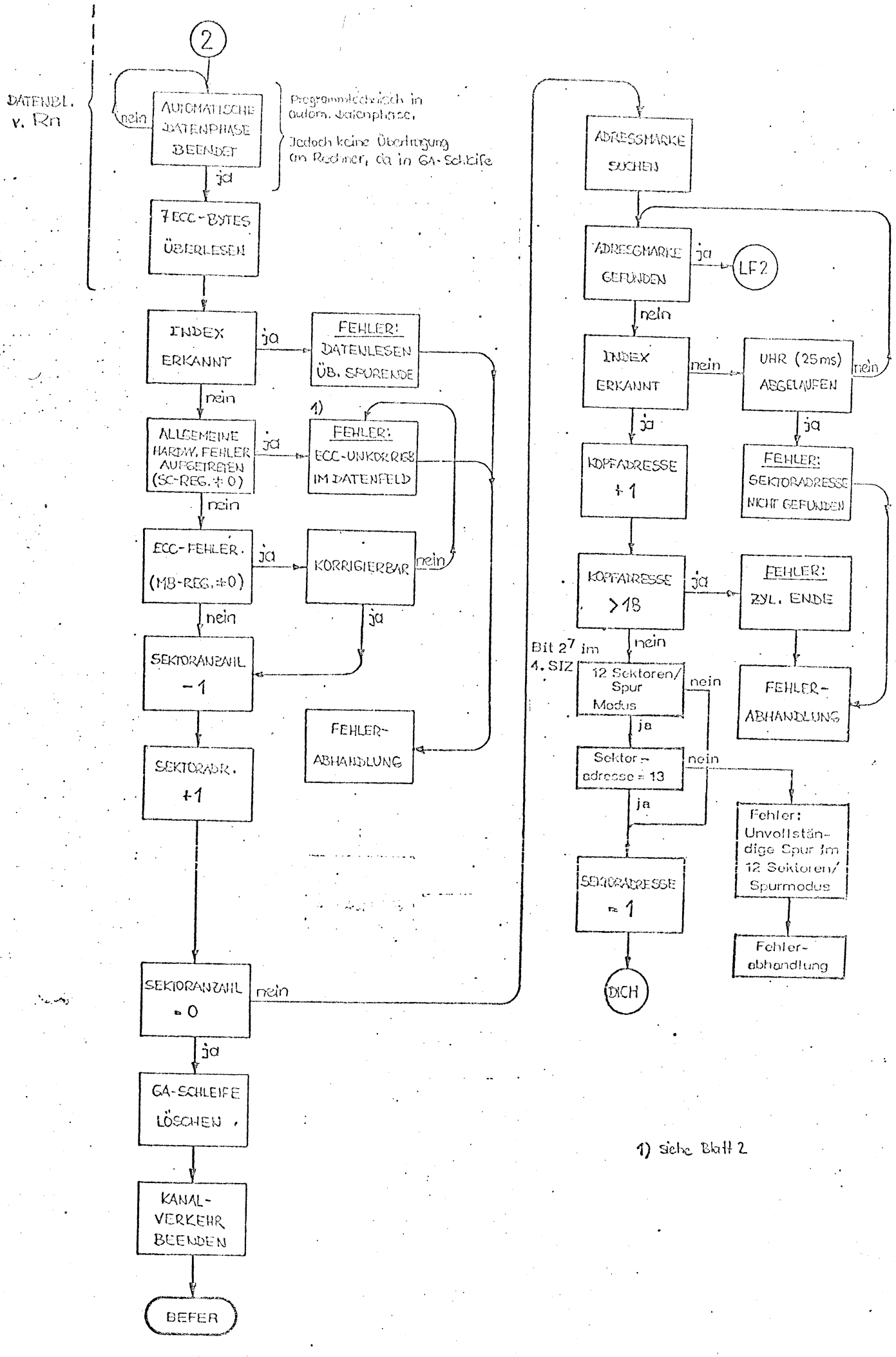
1.13. PRÜFLESEN (BL. 1)



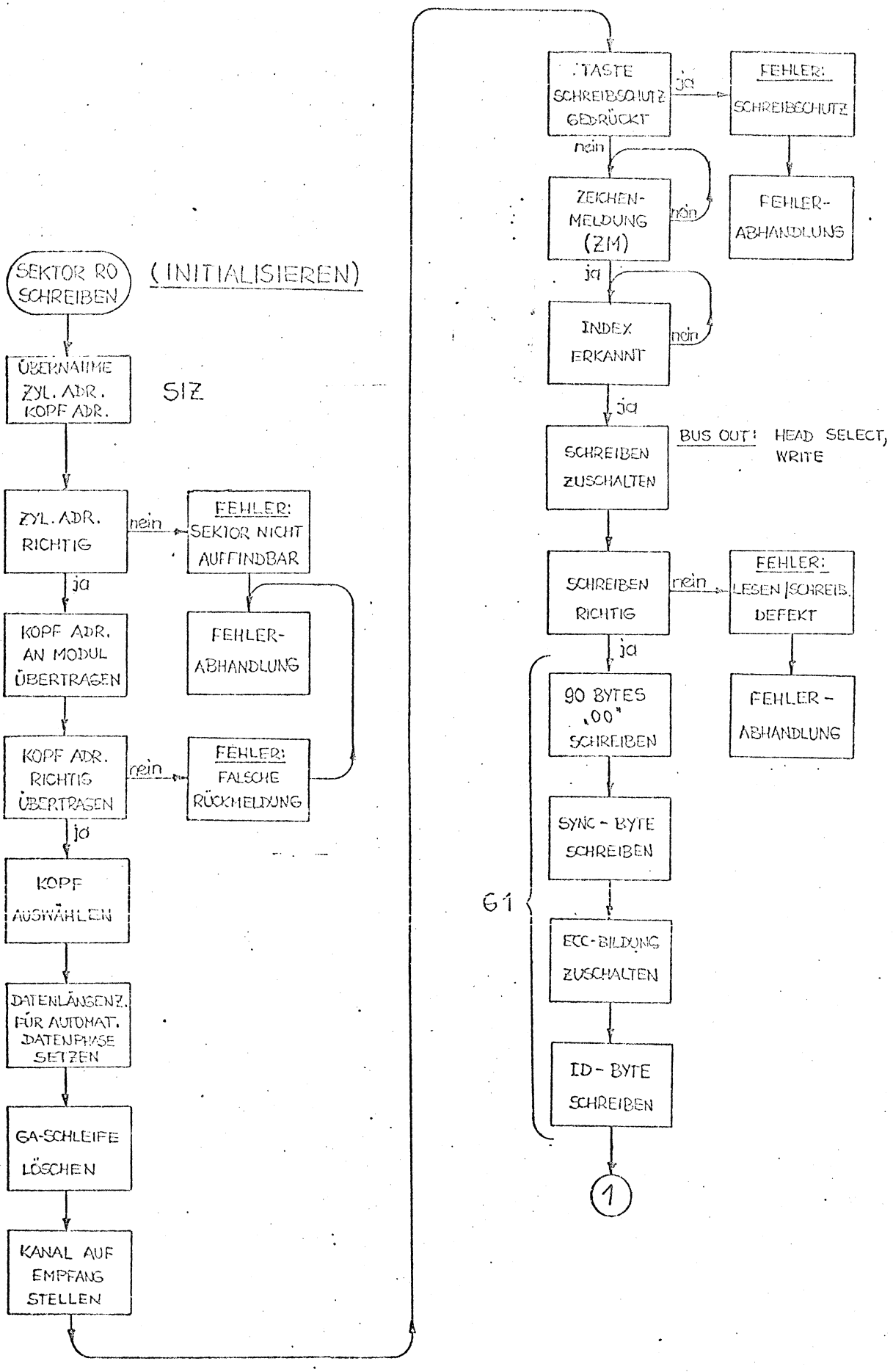
1) Wurde ein Korrekturversuch unternommen, (nur bei MB-Reg ≠ 0), wird zusätzlich im Fehlerbyte 2 das Bit 2^1 gesetzt. („Unkorrigierbarer Fehler nach Korrekturversuch")

2) Daraus folgt: Mit einem Befehl können nur Spuren gelesen werden, die mit aufsteigender Sektoradresse (+1) formatiert wurden.

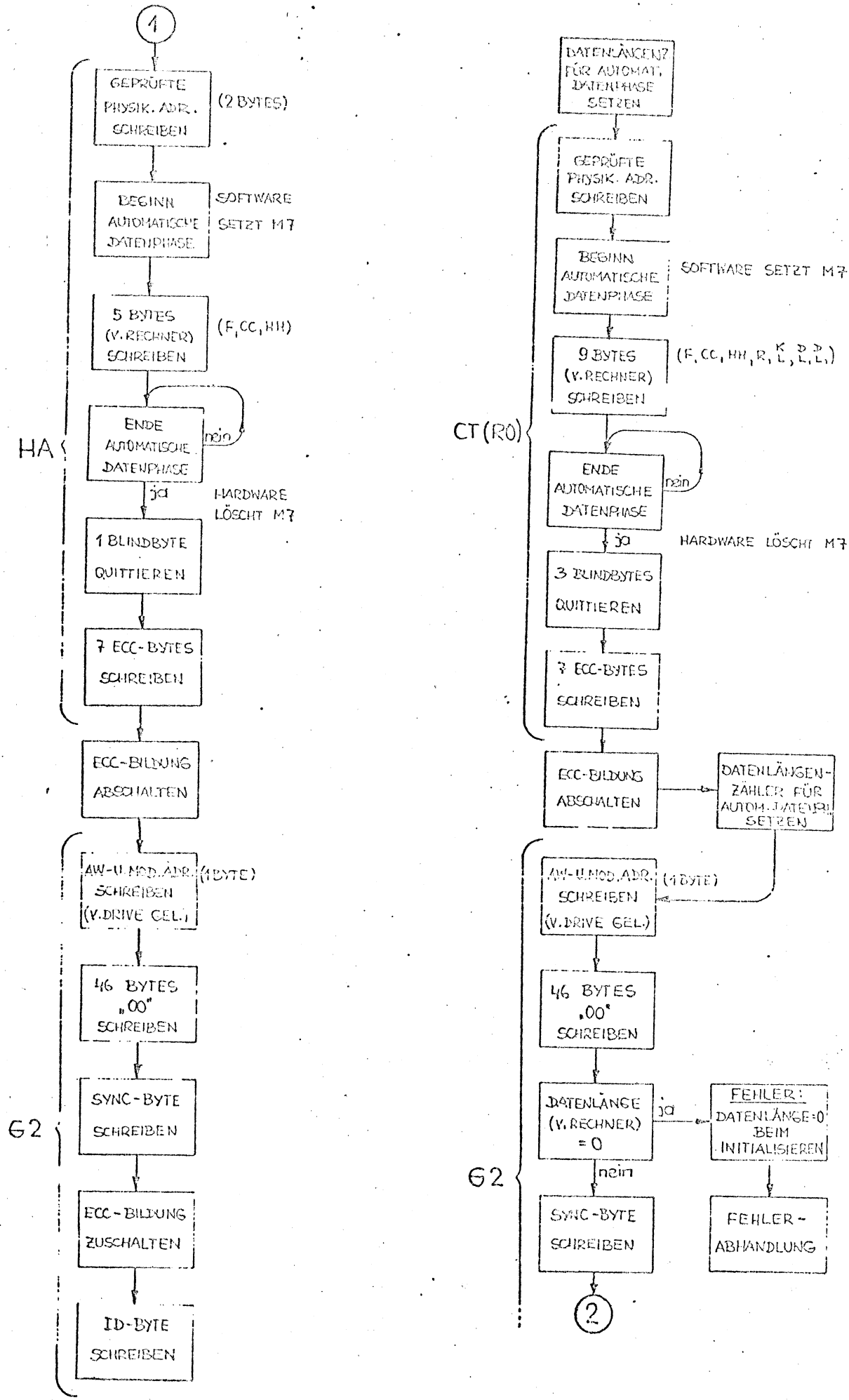
1.13. PRÜFLESEN (BL.2)



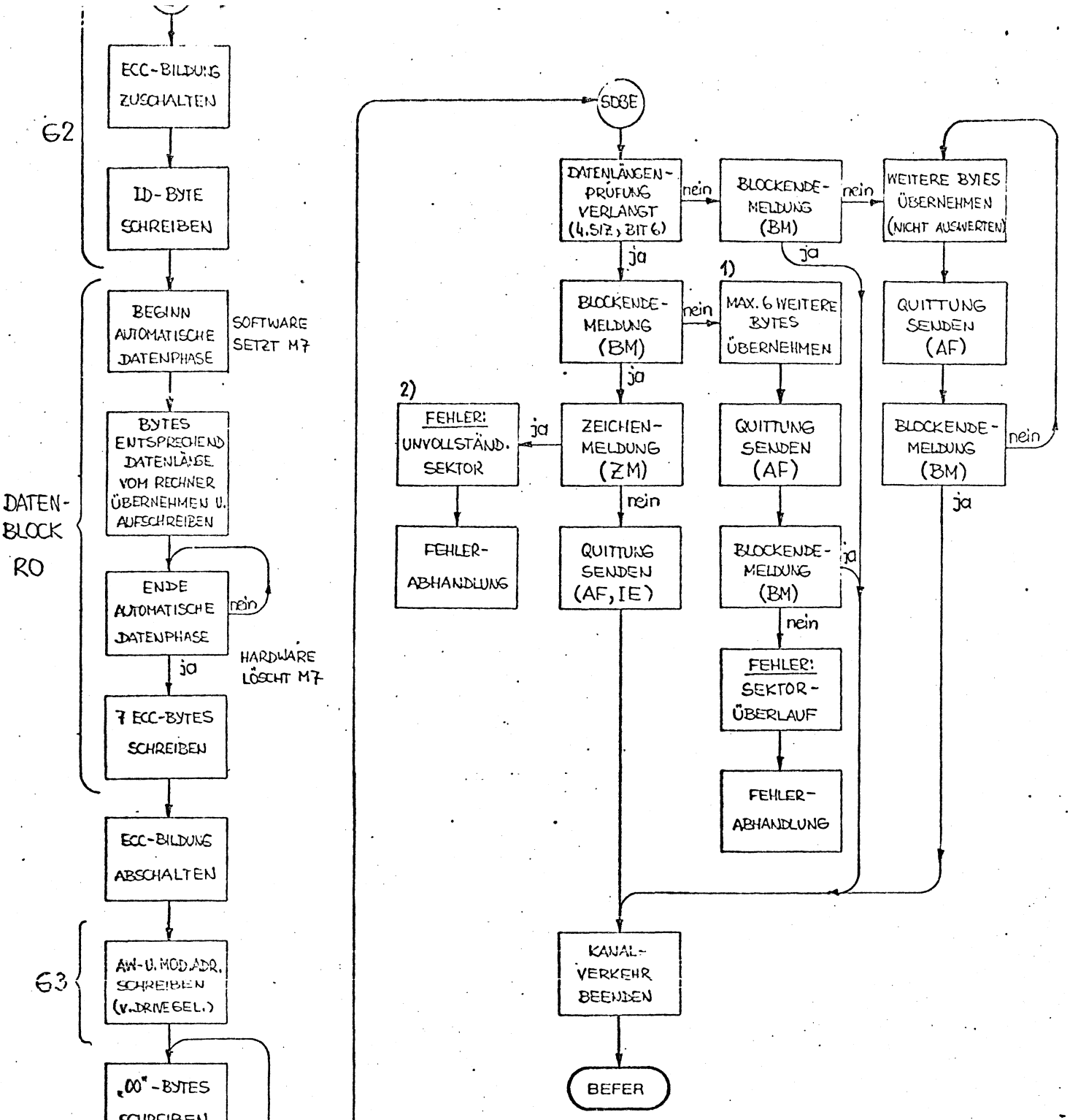
1.13. PRÜFLESEN (BL.3)



1.14. SEKTOR RO SCHREIBEN (BL. 1)



1.14. SEKTOR R0 SCHREIBEN (BL.2)

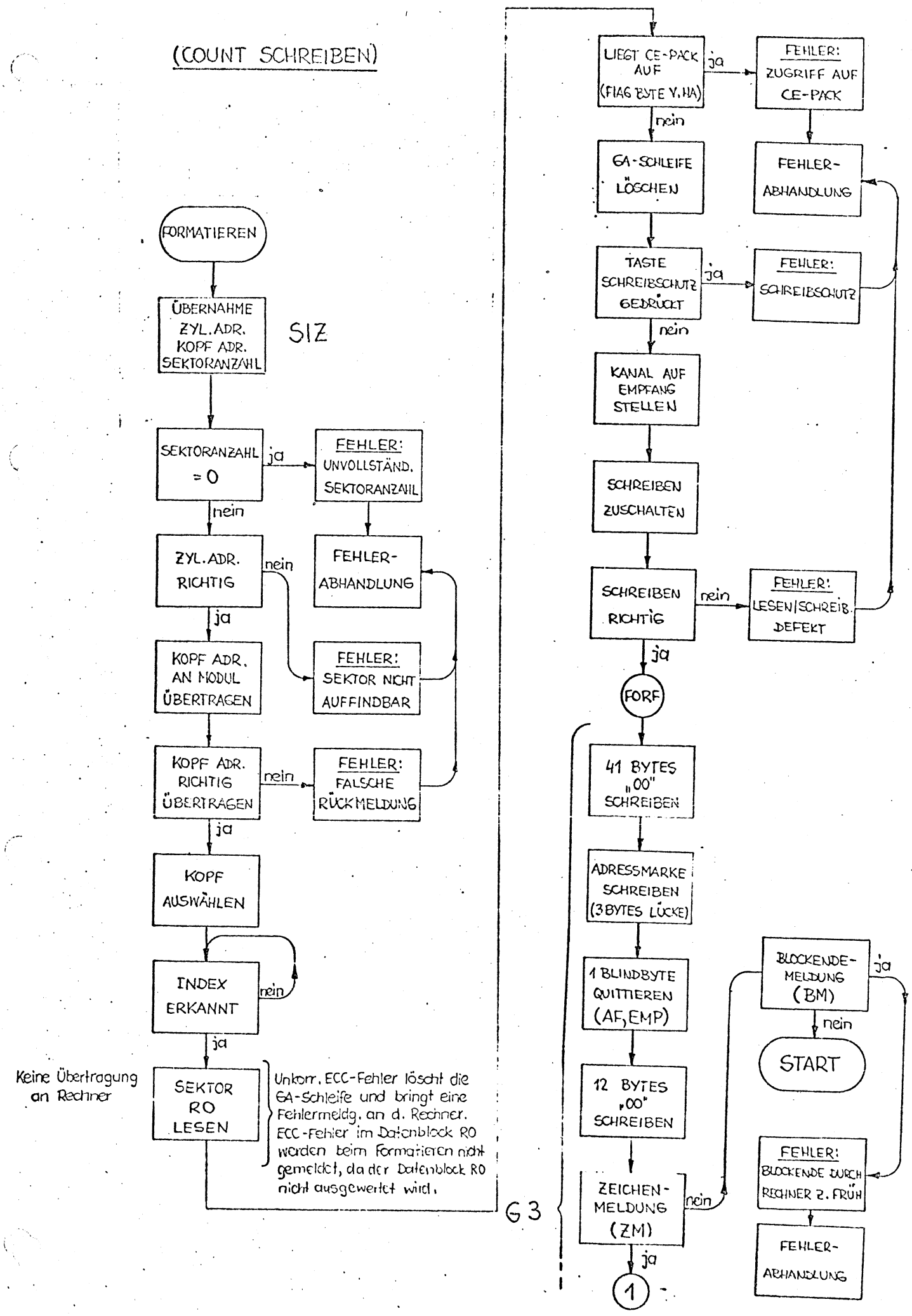


1) Gibt der Rechner mehr als 6 zusätzliche Bytes aus, so bricht das AW mit Fehlereingriff ab.

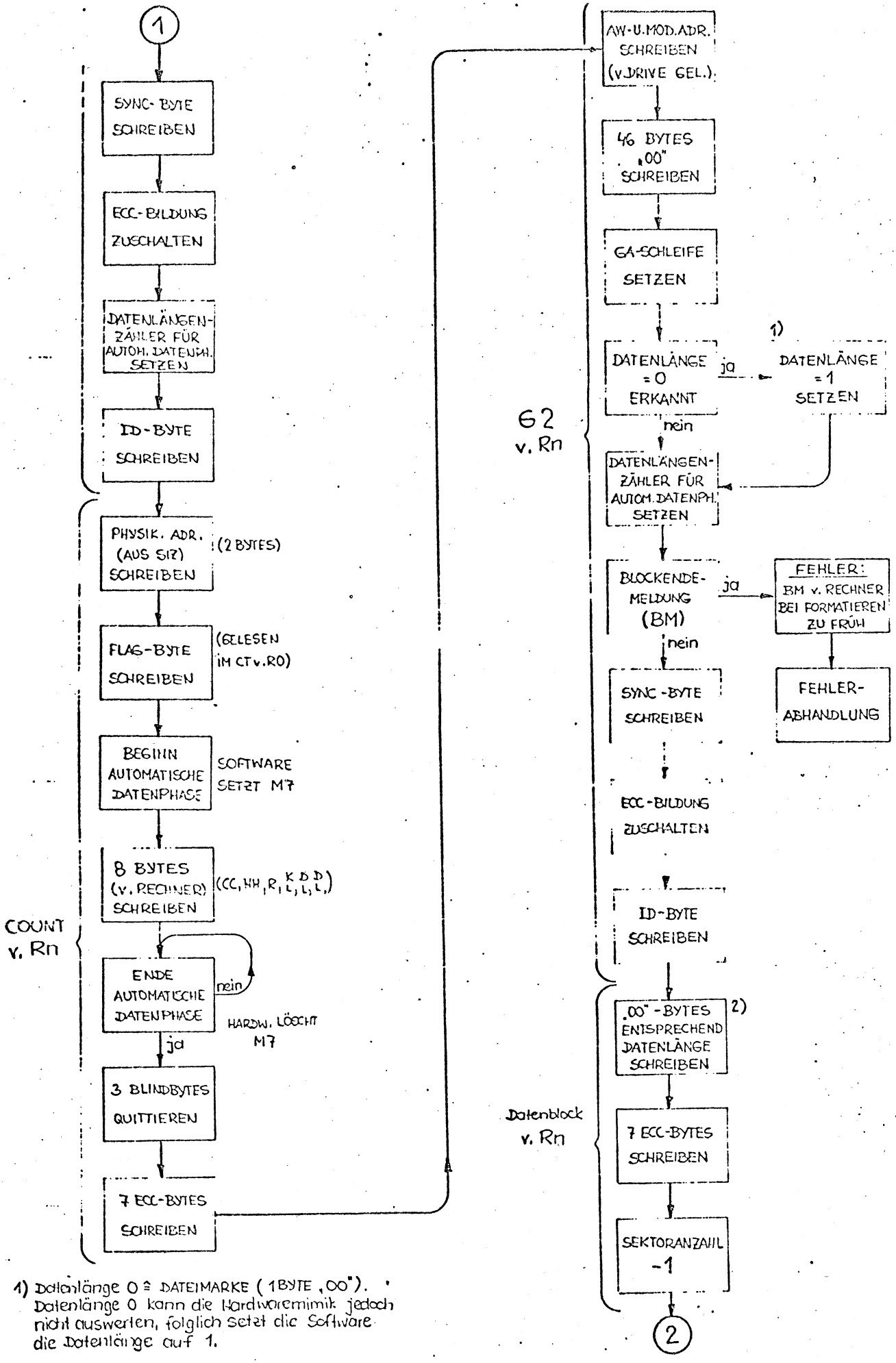
2) Rechner hat vor Sektorende mit BM abgebrochen (d.h. er wollte Teilblock schreiben), ohne die Datenlängenprüfung abzuschalten.

1.14. SEKTOR RO SCHREIBEN (BL.3)

(COUNT SCHREIBEN)

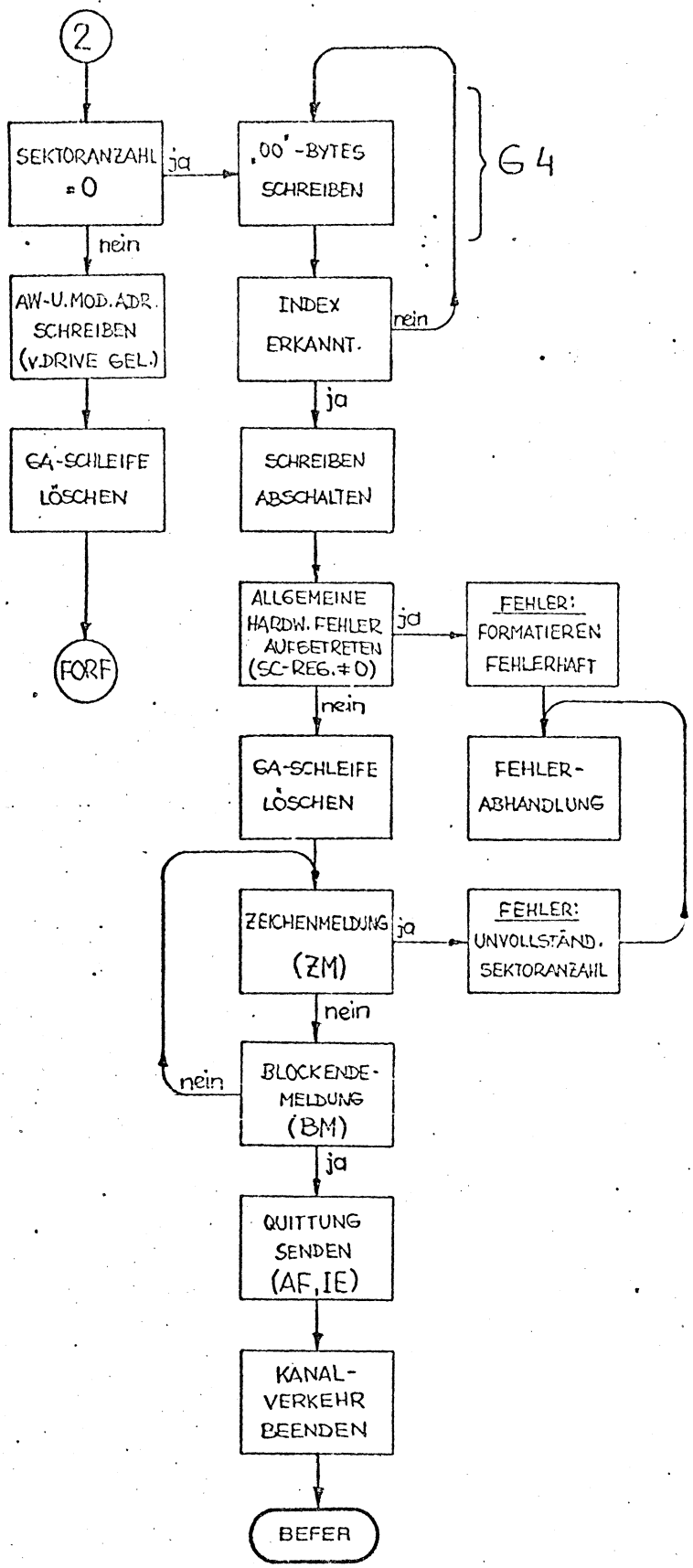


1.15. FORMATIEREN (BL.1)

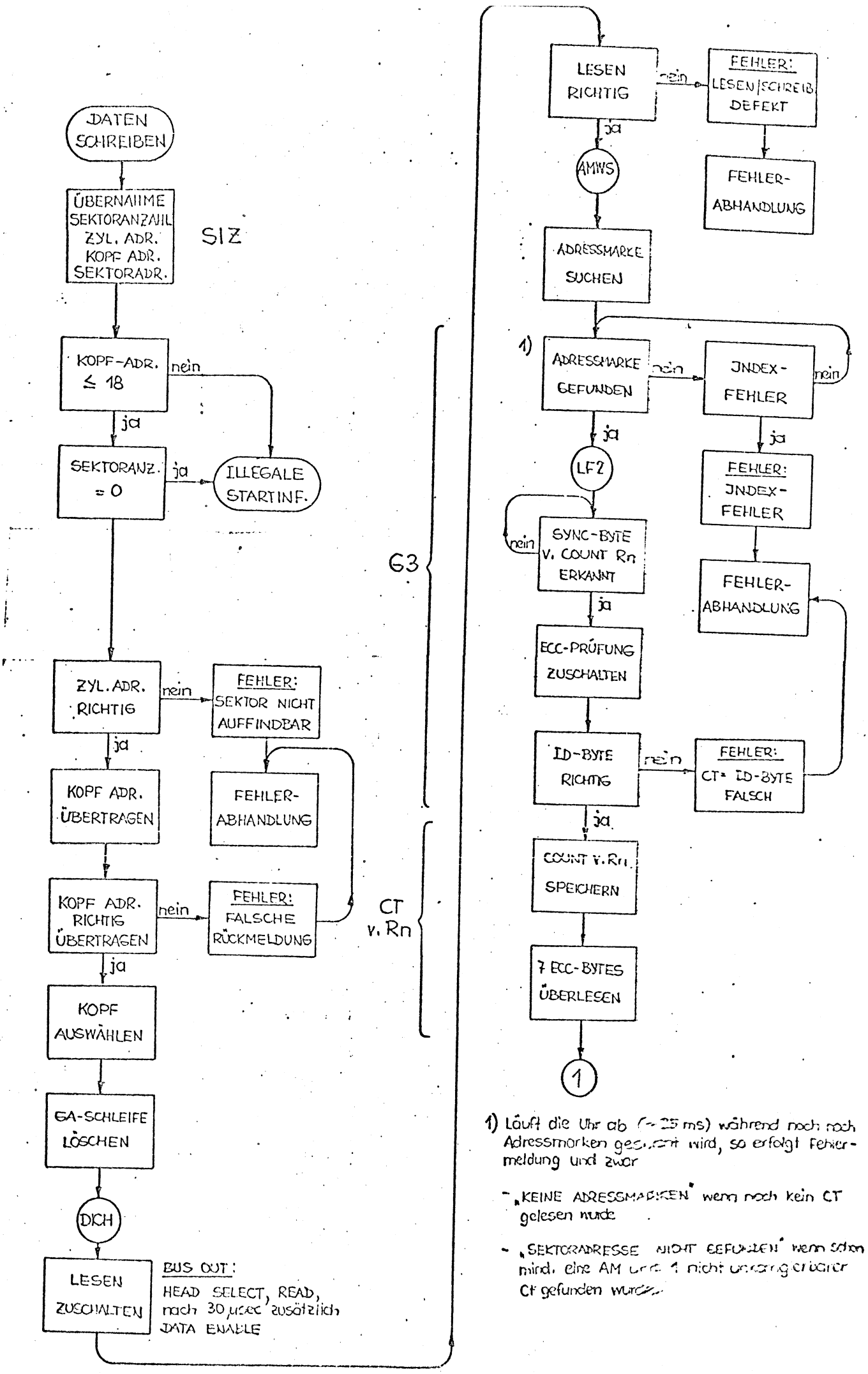


1) Datenlänge 0 ≙ DATEIMARKE (1 BYTE „00"). Datenlänge 0 kann die Hardwaremimik jedoch nicht auswerten, folglich setzt die Software die Datenlänge auf 1.

2) „00"-BYTES schreiben wird programmtechn. in der autom. Datenphase durchgeführt. Da aber GA-Schleife ansteht, werden nur „00"-BYTES aufgeschrieben.



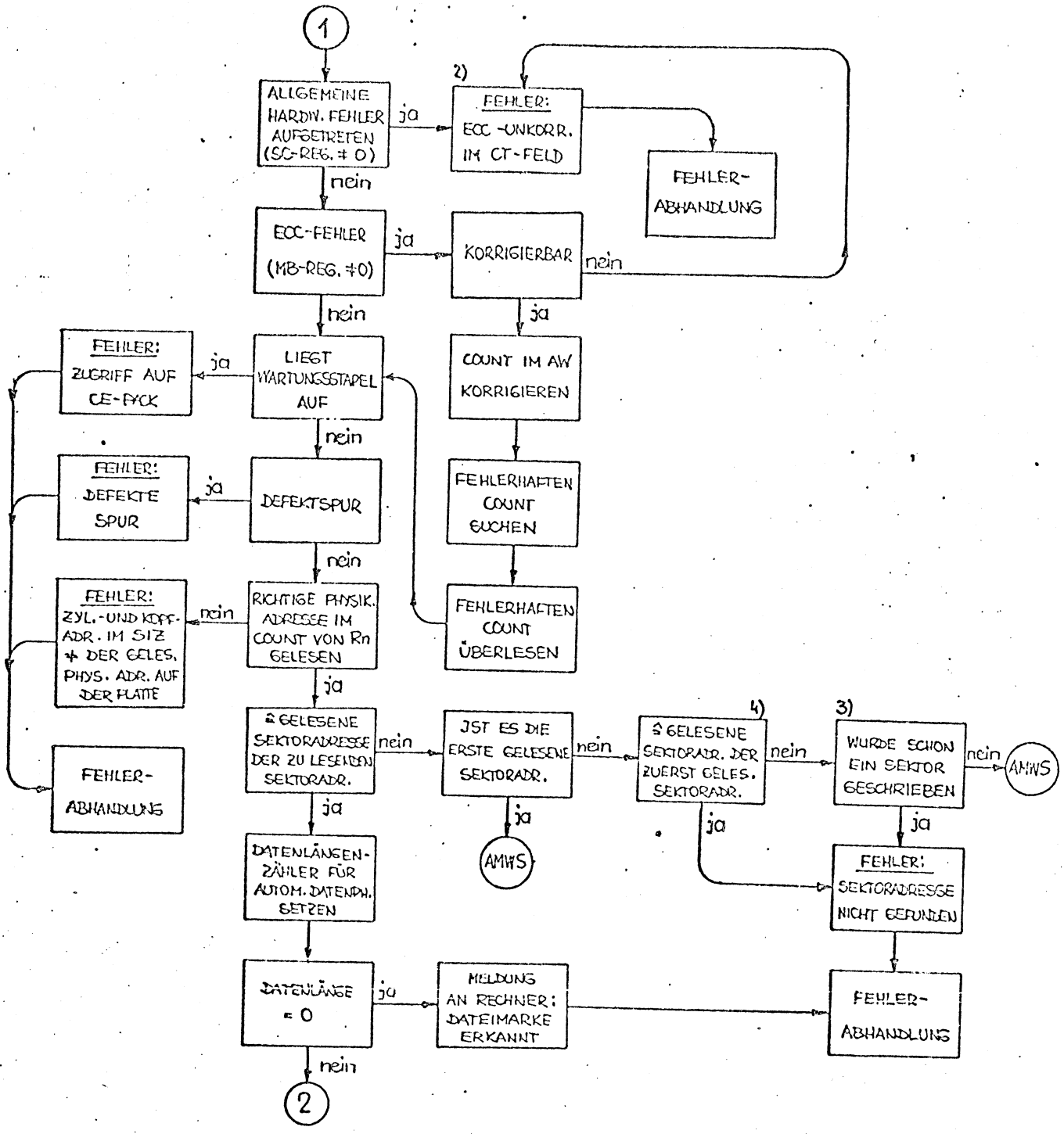
1.15. FORMATIEREN (BL.3)



1) Läuft die Uhr ab (~ 25 ms) während noch nach Adressmarken gesucht wird, so erfolgt Fehlermeldung und zwar

- „KEINE ADRESSMARKEN" wenn noch kein CT gelesen wurde
- „SEKTORADRESSE NICHT GEFUNDEN" wenn schon mind. eine AM und 1 nicht unbedingt erforder CT gefunden wurde

1.16. DATEN SCHREIBEN (BL 1)

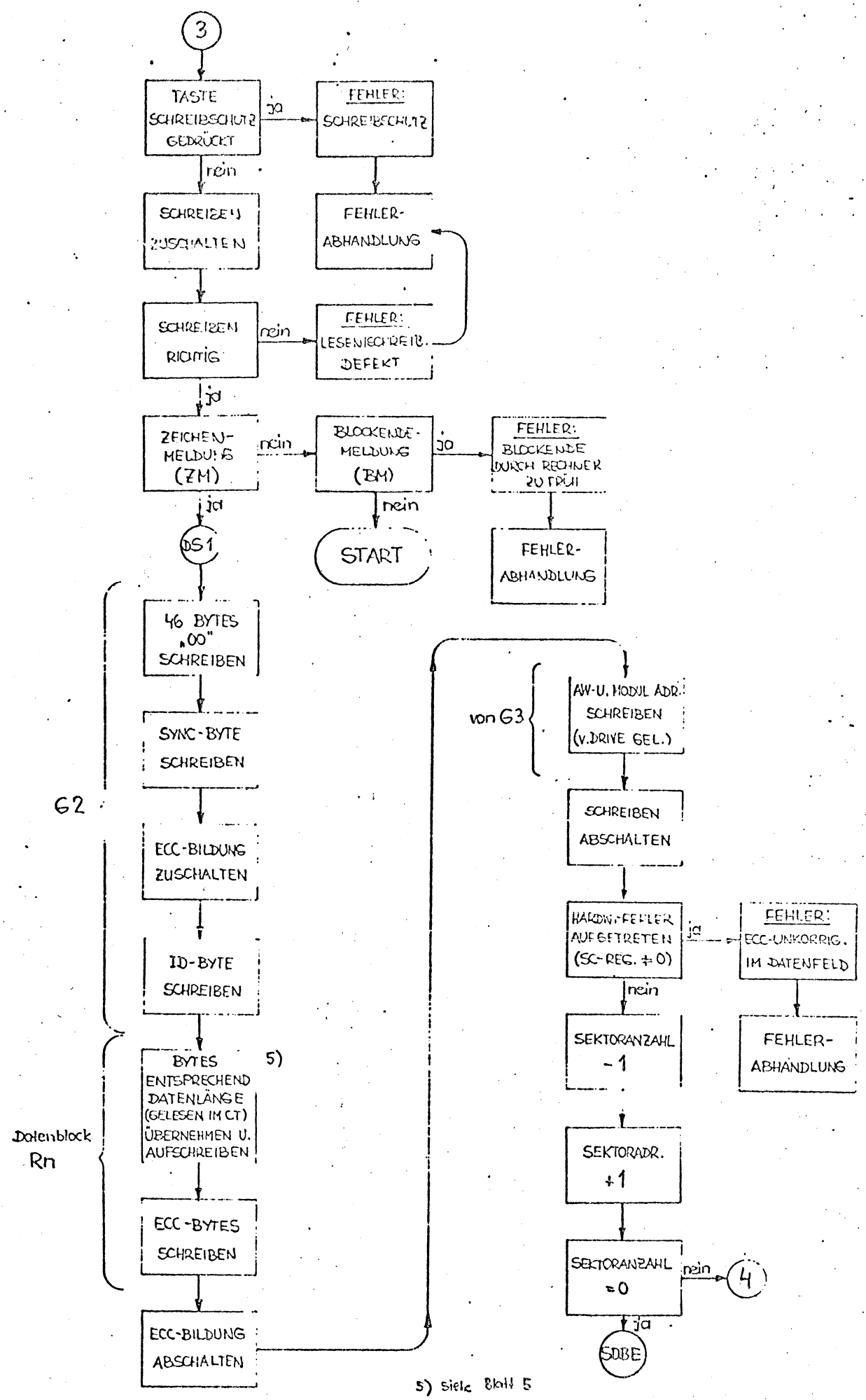


2) Wurde ein Korrekturversuch unternommen (nur bei MB-Reg. ≠0), wird zusätzlich im FB 2 das Bit 2⁴ gesetzt. („UNKORRIGIERBARER FEHLER NACH KORREKTURVERSUCH").

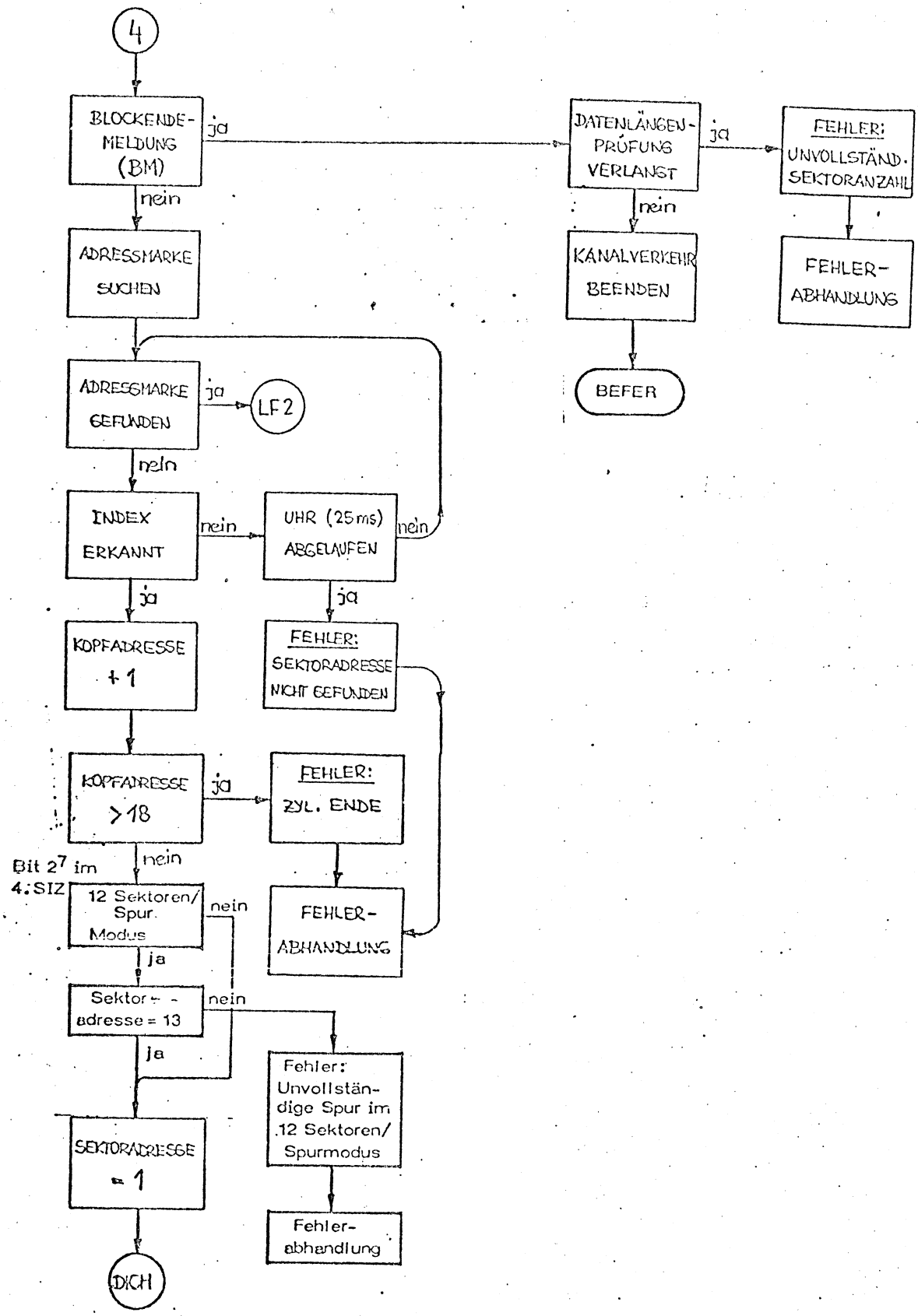
3) Daraus folgt: Mit einem Befehl können nur Spuren beschrieben werden, die mit aufsteigender Sektoradresse (+1) formatiert wurden.

4) Es muß angenommen werden, daß bereits 1 Plattenumdrehung lang gesucht wird.

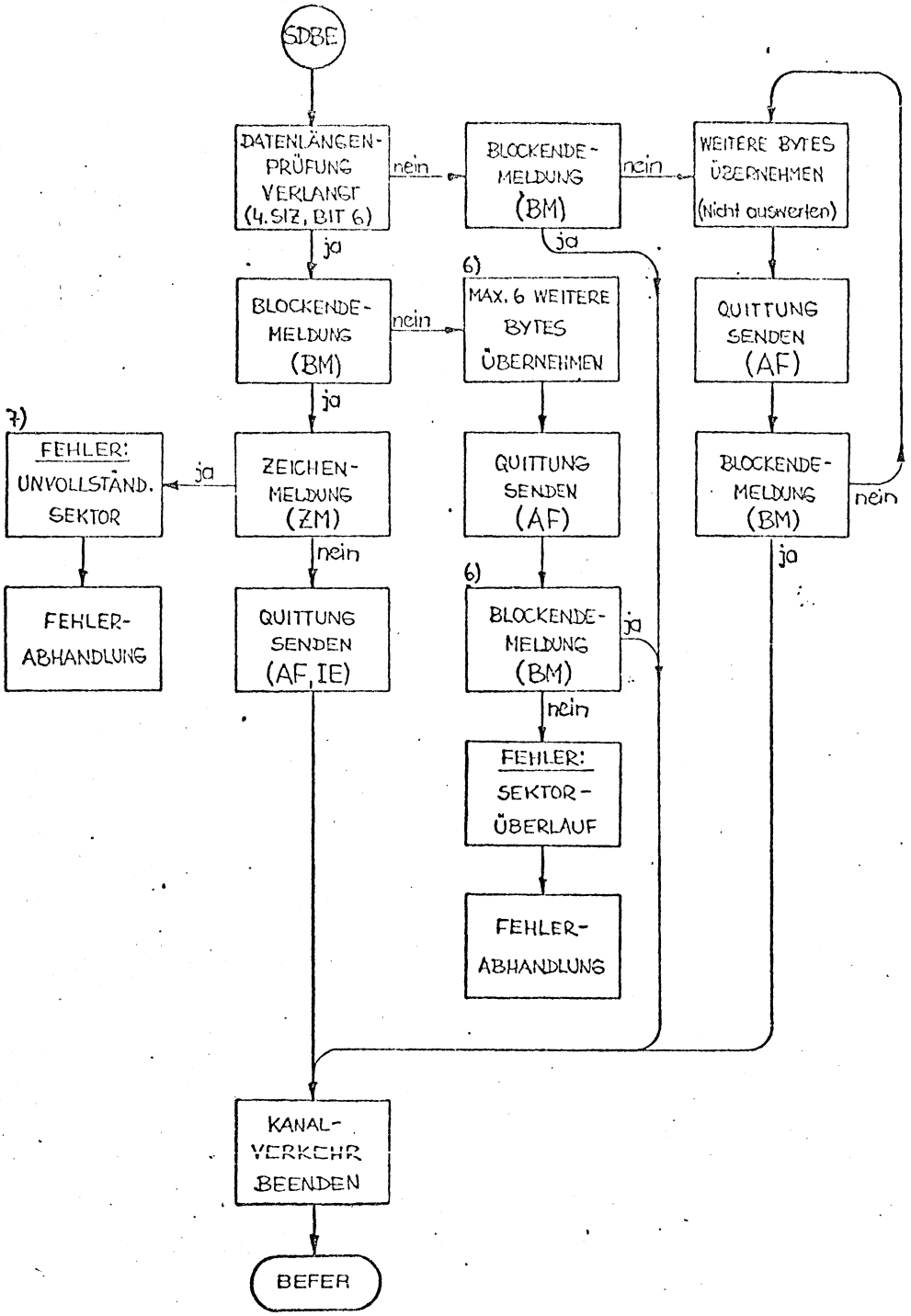
1.16. DATEN SCHREIBEN (BL.2)



1.16. DATEN SCHREIBEN (BL.3)



1.16. DATEN SCHREIBEN (BL.4)



5) Beim „FORMATIEREN" wird die Datenlänge (DL) festgelegt und steht im Count (Ct) von $R_1 \ldots R_n$.

6) Gibt der Rechner mehr als 6 zusätzliche Bytes aus, so bricht das AW mit Fehlereingriff ab.

7) Rechner hat vor Sektorende mit BM abgebrochen (d.h. er wollte Teilblock schreiben), ohne die Datenlängenprüfung abzuschalten.

1.16. DATEN SCHREIBEN (BL.5)

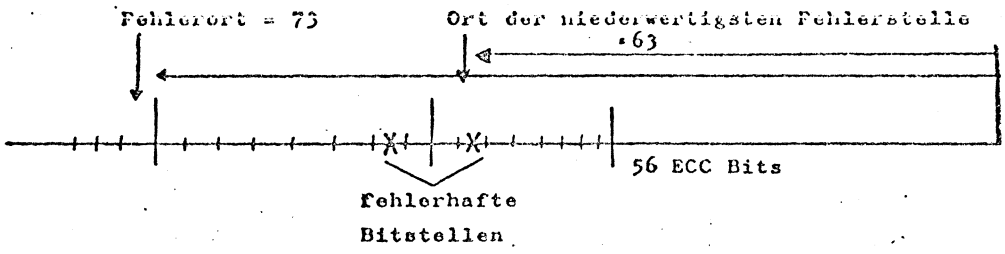
1.17. ECC-Korrektur

Die ECC-Bytes werden folgendermaßen gebildet: Die als Polynom aufgefaßte Folge der Datenbits wird durch ein Generatorpolynom (definiert durch die rückgekoppelten Polynomgeneratoren P_0 - P_3) dividiert. Der dabei entstehende Rest wird den Datenbits angefügt. Dies sind die 56 ECC-Bits. Inclusive der ECC-Bytes ist das Datenpolynom jetzt ohne Rest durch das Generatorpolynom ($P_{(x)}$) teilbar (Begründung siehe ECC-Verfahren). Dies wird beim Lesen ausgenützt.

Bei der Decodierung wird das gesamte Datenpolynom wieder durch $P_{(x)}$ dividiert. Bei unverfälschten Daten enthalten alle Stellen von P_0 - P_3 Nullen d.h. MB = 0. Sind irgendwelche Bits verfälscht worden, ist MB ≠ 0 und P_0 (nur eine Rückkopplung) enthält das Fehlermuster. Ist durch max. 22 Shifts von P_0 nicht zu erreichen, daß die rechte Hälfte von P_0 = 0, so ist der Fehler unkorrigierbar. Konnte die rechte Hälfte von P_0 zu Null gemacht werden, so steht das Fehlermuster in der linken Hälfte von P_0. Es muß nun bei einem korrigierbaren Fehler möglich sein durch Shiften von P_1 - P_3 dieses Fehlermuster auch in P_1 - P_3 zu erhalten. Dies muß nicht gleichzeitig geschehen. Die größte Periode von P_1 - P_3 ist 89 Shifts. Konnte also nach 89 Shifts keine Übereinstimmung gefunden werden, so ist der Fehler nicht korrigierbar.

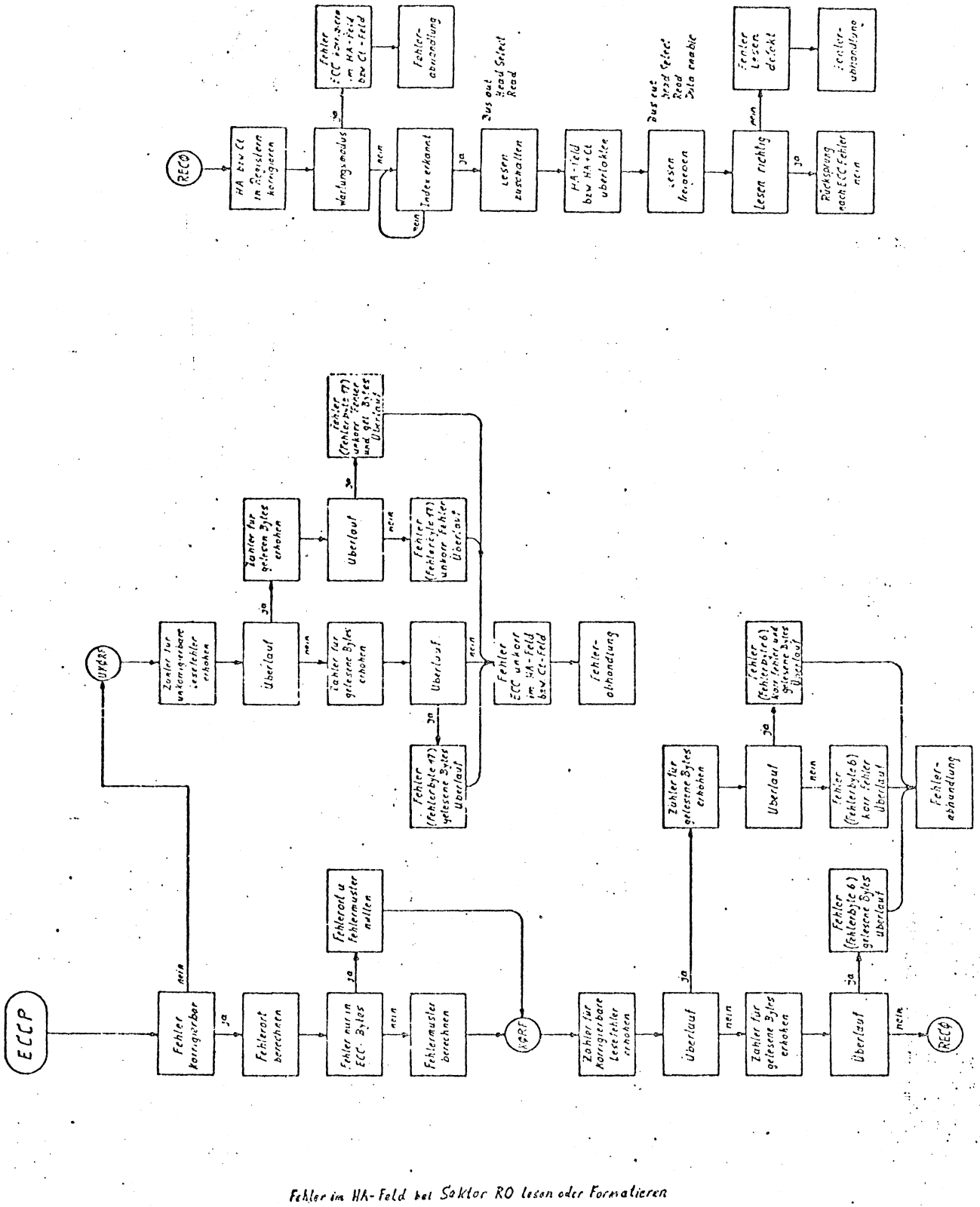
Aus der Anzahl der benötigten Shifts bei P_0 - P_3 kann der Fehlerort (Abstand des Fehlermusters in Bit vom Informationsende ab) errechnet werden.

Bei diesem Verfahren erhält man als Fehlerort im Accu (PE HC HE) den Ort der niederwertigsten Fehlerstelle vom Ende der Information plus 10. Es wird mit 1 beginnend gezählt, deswegen noch plus 1.
Insgesamt also plus 11. Das ist auch die Vorbesetzung des Accus mit 'B0'. Die Linksverschiebung um 4 Bit ist aus programmtechnischen Gründen wegen der nachfolgenden Division durch 8 gemacht worden.

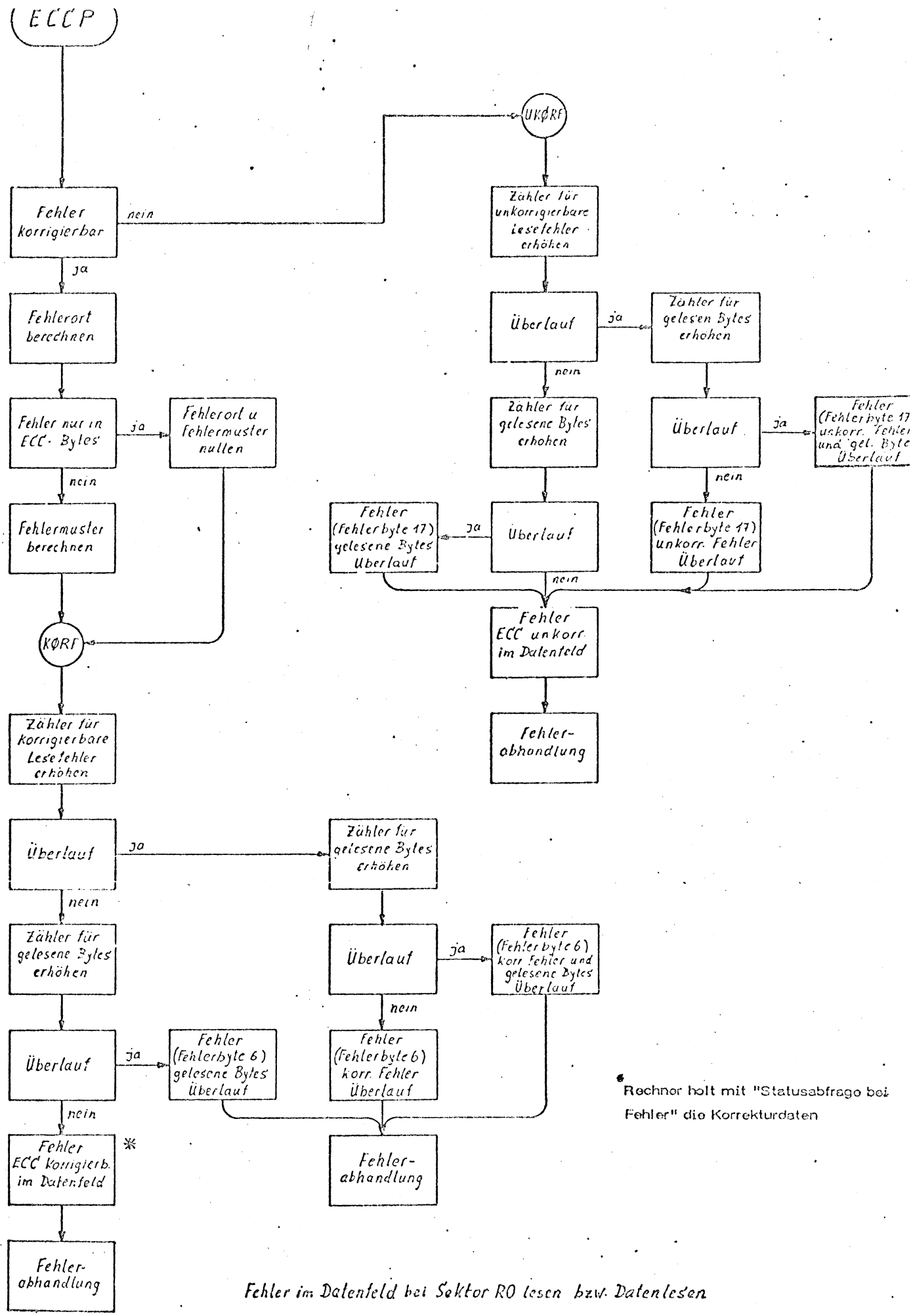


Fehlerort = 73 Bit, d.h. 9 Byte . Der Rechner bekommt den Fehlerort ab Ende Daten also 9-6 = 3

Da der Rechner den Fehlerort in Byte verlangt, muß der Fehlerort in Bit noch durch 8 geteilt werden. Der Rest, der sich bei dieser Division ergeben kann, dient dazu, das in P_0 stehende Fehlermuster bytegerecht zu verschieben. Schließlich werden die3Fehlermusterbytes aus P_0 nach ME geladen und im Datenspeicher abgelegt. Je nachdem welche Information (HA, Ct oder Daten) zu korrigieren ist, wird im AW korrigiert, oder die Daten zum Rechner übertragen (siehe Einzelbeschreibung im Strukturdiagramm).

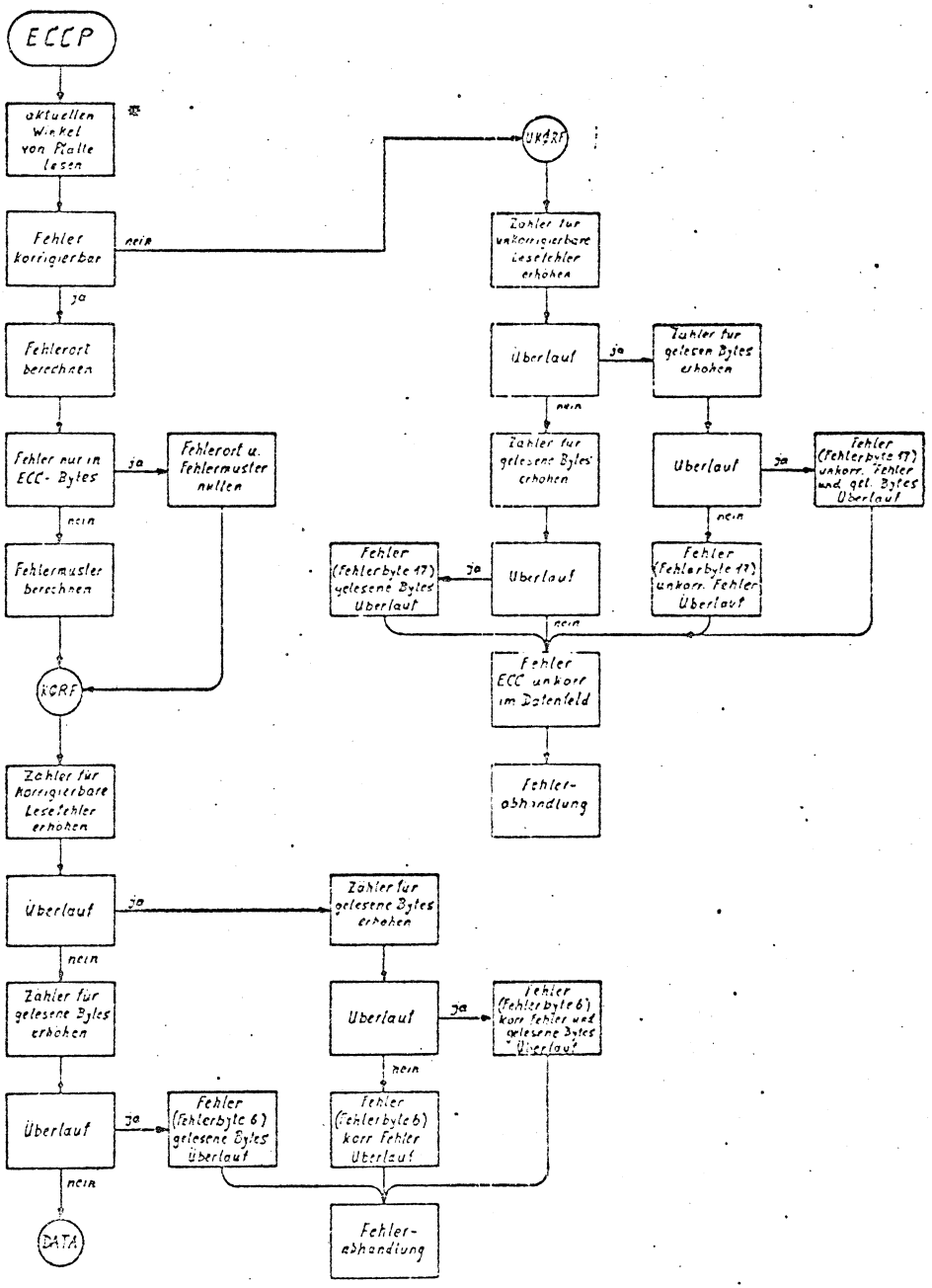


Fehler im HA-Feld bei Sektor RO lesen oder Formatieren

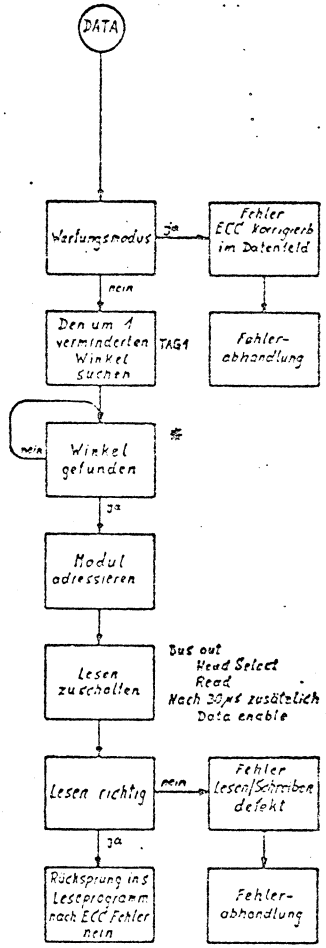


* Rechner holt mit "Statusabfrage bei Fehler" die Korrekturdaten

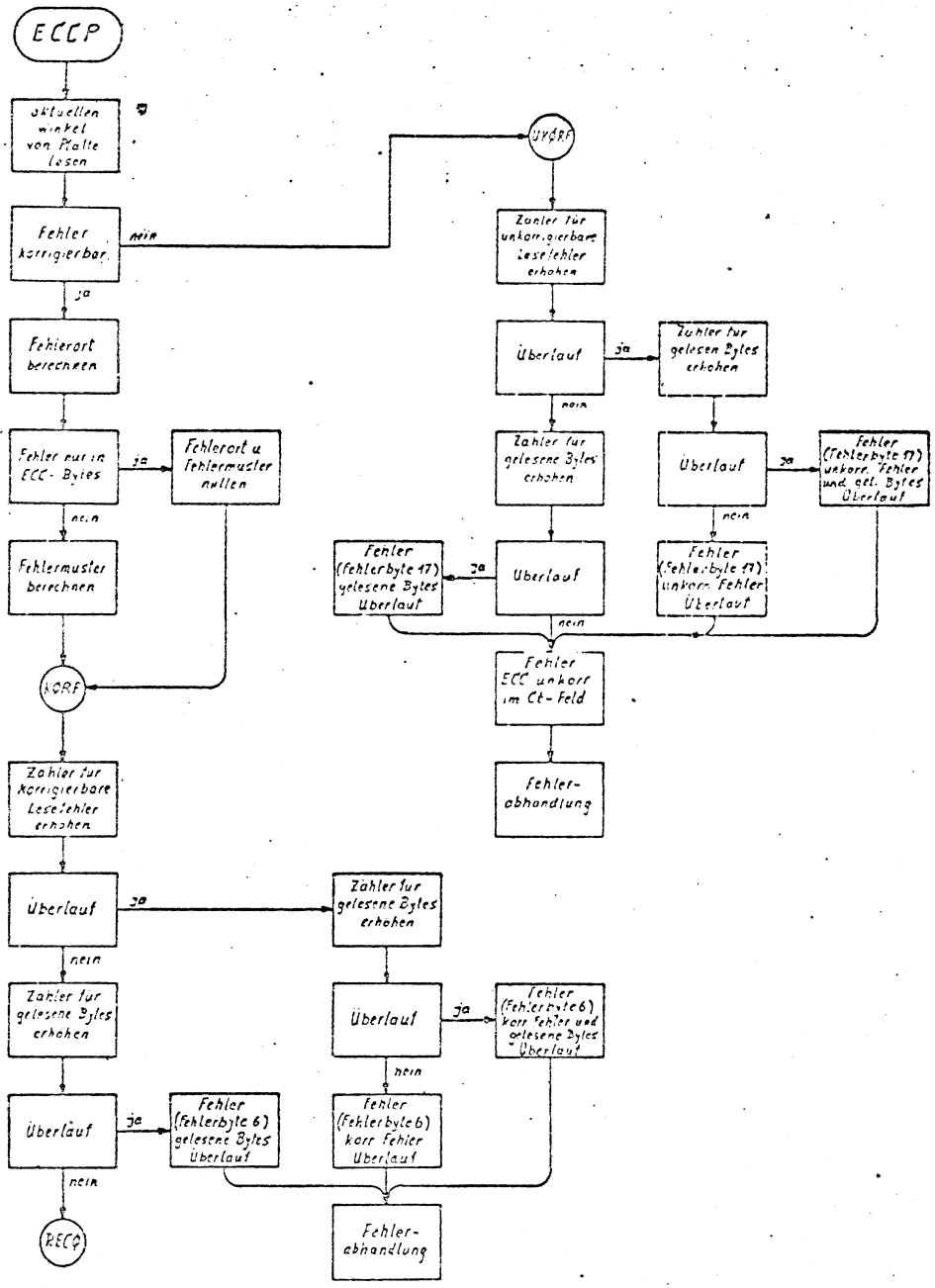
Fehler im Datenfeld bei Sektor RO lesen bzw. Daten lesen



* Die Platte ist in 128 Winkel eingeteilt. Dies dient dem Wiederfinden einer bestimmten Stelle ohne nochmaliges Lesen, da ein Befehl an den Drive "Winkelsuchen" existiert. Bei Erreichen dieses Winkels wird ein Interrupt gegeben.

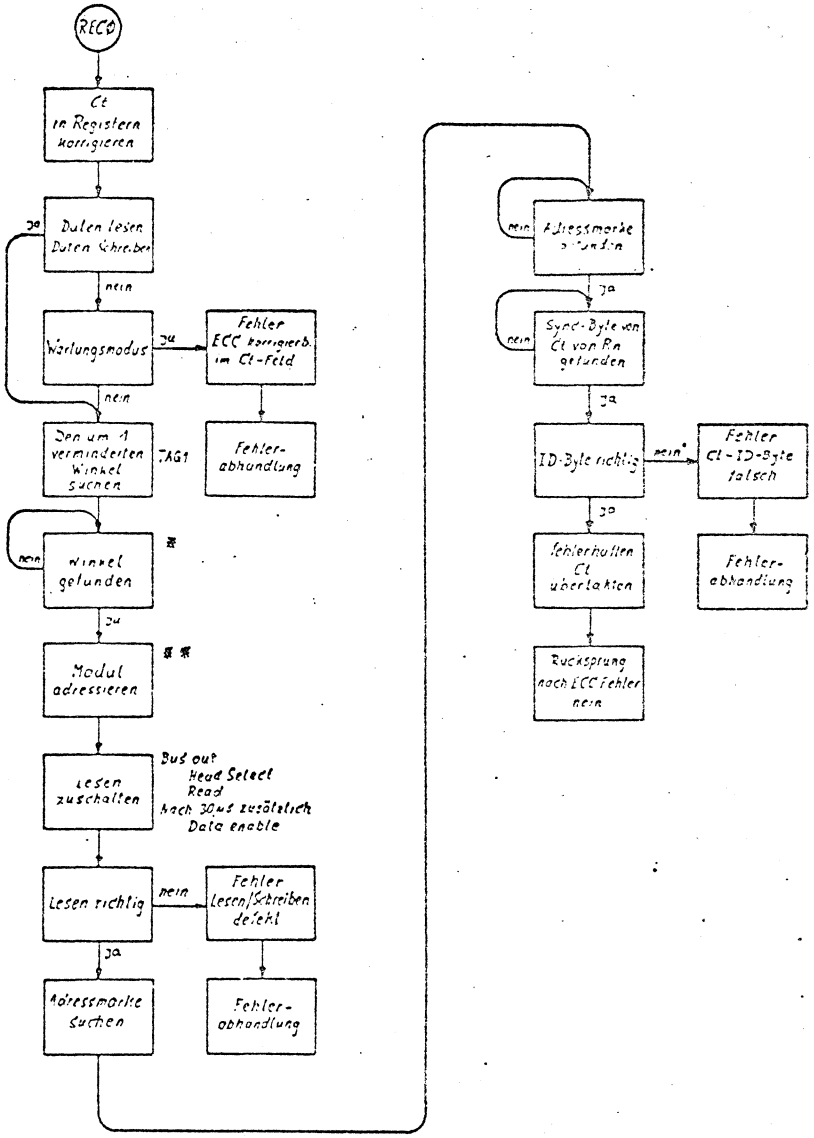


* Der Drive meldet Interrupt (Record-Ready-Interrupt), wenn der gesuchte Winkel gefunden ist. Durch die Interrupt-Abfrage (TAG 2) wird die Adressierung zerstört.



Fehler im Ct-Feld bei Count lesen bzw. Daten Schreiben
bzw. Daten Lesen
bzw. Prüflesen

* Die Platte ist in 128 Winkel eingeteilt. Dies dient dem Wiederfinden einer bestimmten Stelle ohne nochmaliges Lesen, da ein Befehl an den Drive "Winkel suchen" existiert. Bei Erreichen dieses Winkels wird ein Interrupt gegeben.



* Der Drive meldet Interrupt (Record-Ready-Interrupt) wenn der gesuchte Winkel gefunden ist. Durch die Interrupt-Abfrage (TAG 2) wird die Adressierung zerstört.

** Die Adressierung des Moduls wurde zerstört, da das Finden des Winkels per Interrupt gemeldet wird und die Abfrage des Interrupts TAG 2) die Adressierung zerstört.

Zeichnungs-Nr.:
25.2201.600-00 FD

ÄNDERUNGSSTAND (ae)			Blatt 1
Reg. 2	ae	Auszutauschende bzw. zu ergänzende Seiten (·)	Datum
	-	-	10.7.76
	A	Auszutauschen: 4.3.2-340, 350, 360, 400, 410, 420,	6.4.77
		440, 650, 700	"
	A	Einzufügen: 4.3.2-341, 701	"

INHALTSVERZEICHNIS

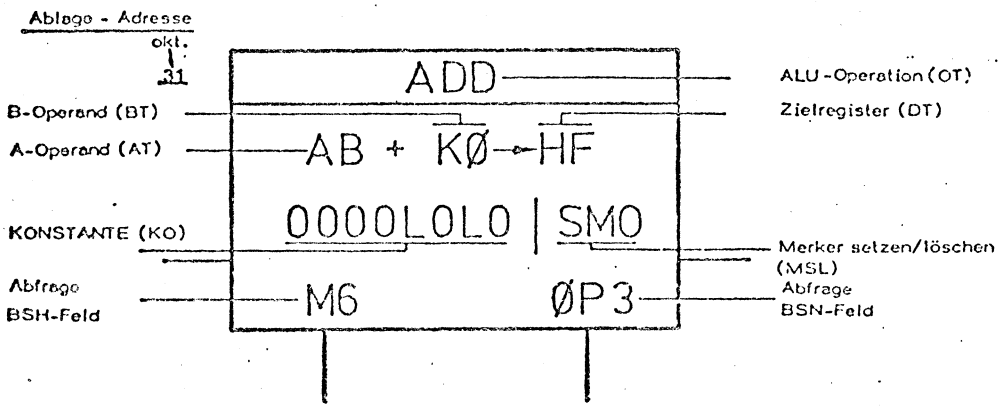
1. FLUSSDIAGRAMM

1.1. Erläuterungen

1.1.1. Symbolerklärung

Da es praktisch unmöglich ist, anhand des Assemblates den Befehlsablauf schnell zu überschauen, wurde das vorliegende Flußdiagramm erstellt.

Darin ist jeder Befehl in folgender Form dargestellt:



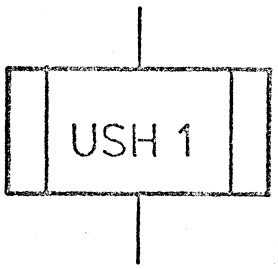
Die Folgeadresse ist aus der Ablageadresse des folgenden Befehls ersichtlich.

Beispiel:
- Addiere ⟨AB⟩-Reg. mit der Konstanten und speichere das Ergebnis im HF-Register.
- Setze den Merker 0
- Abfrage: Sitzt der Merker 6 und ist im OP-Reg. das Bit 2^3 = "L"? (entsprechend M6 und OP3 erfolgt Errechnung der Folgeadresse, d.h., vier Ansprungsmöglichkeiten)

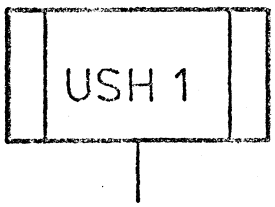
Erklärung zu ALU-Operationen:
+ Addition
- Subtraktion
* Konjunktion
V Disjunktion
⩛ Exclusiv Oder

Vor Befehlen, die von mehreren Stellen angesprungen werden, ist ein Konnektor angegeben.
Die Definitionsadressen und die Ansprünge von Konnektoren sind in der Konnektoren-Liste angegeben.
Bei den Konnektorenansprüngen im Flußdiagramm ist jeweils die Definitionsadresse und -seite angegeben.

Unterprogrammdurchläufe sind folgendermaßen dargestellt:



Das Unterprogramm beginnt mit dem gleichen Symbol:



Mit dem indizierten Sprung (AMA) kann abhängig vom Registerstand ein Befehl aus einem zusammenhängenden Bereich angesprungen werden. Diese Bereiche heißen Listen. Sie werden folgendermaßen dargestellt:

SIZSP

0	A	KA	← PL	Spr.	← ENSIZ		64
1	A	KA	IS	Spr.	UPSIZ		65
2	A	KA	MA	Spr.	UPSIZ		66
3	A	KA	HC	Spr.	UPSIZ		67
4	A	KA	HD	Spr.	UPSIZ		70
5	A	KA	MS	Spr.	UPSIZ		71
6	A	KA	HE	Spr.	UPSIZ		72
7	A	KA	PE	Spr.	UPSIZ		73

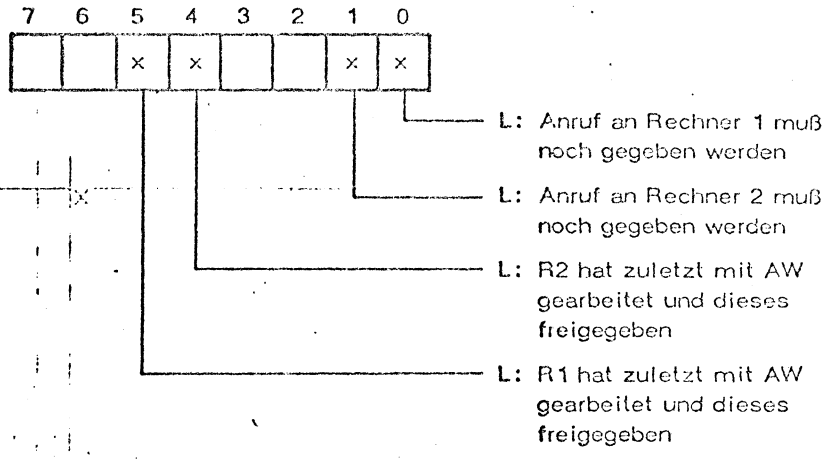
Die erste Spalte ist die relative Adresse in der Liste; am Schluß ist die absolute Oktaladresse angegeben. Dazwischen ist der Befehl aufgeführt und zwar nur die Bestandteile, die relevant sind.

0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
FB 0	FB 1	FB 2	FB 3	FB 4	FB 5	FB 6	FB 7	FB 8	FB 9	FB 10	FB 11	FB 12	FB 13	FB 14	FB 15
20 FB 16	21 FB 17	22 LW-R1	23 Verreg. R2	24 LW-R1	25 Reserv. R2	26 Weiche 2	27 LW-Angabe	30 Zuerst gelesene Sektor-Adr.	31 PE	32 HC	33 HE	34 ØP	35 HM	36 HN	37 AB
40 HF	41 HG	42 HI	43 HØ	44 HP	45 Anruf R1	46 Byte R2	47 Modul-Adresse	50 Winkel	51 Interrupt Byte	52 aktuelle Sektor-Adr.	53	54	55 ← Modul A	56	57
60	61	62	63 Modul A	64	65	66	67	70	71	72	73 Modul B	74	75	76	77
100	101	102	103	104	105	106 Modul C	107	110	111	112	113	114	115	116	117
120	121	122	123 Modul D	124	125	126	127	130	131	132	133	134 Modul E	135	136	137
140	141	142	143	144	145	146	147	150 Modul F	151	152	153	154	155	156	157
160	161	162	163 Modul G	164	165	166	167	170	171	172	173	174 Modul H	175	176	177
200	201 Modul H	202	203	204	205	206 Z2	207 Z1	210 Sektor-Anzahl	211 AW-Adresse	212	213	214	215	216	217
220	221	222	223 Status für	224	225	226 R1	227	230	231	232	233	234	235	236	237
240	241	242	243 Status für	244	245	246 R2	247	250	251	252	253	254	255	256	257
260	261	262	263	264	265	266	267	270	271	272	273	274	275	276	277
300	301	302	303	304	305	306	307	310	311	312	313	314	315	316	317
320	321	322	323	324	325	326 Datenbereich für	327	330	331 HCL	332	333	334	335	336	337
340	341	342	343	344	345	346 Datenbereich für	347	350	351 Diagnostik-	352	353 Routinen	354	355	356 Rout.Nr.	357 3/6
360 Fehler	361 Folge-Routine	362 1. MA	363 2. DR Parameter für	364 3. PL Diagnost	365 4. IS	366 5. HE A. Rout	367 6. PE	370	371 Ergebnisse	372	373 von Diagnostik-Routinen	374	375	376	377

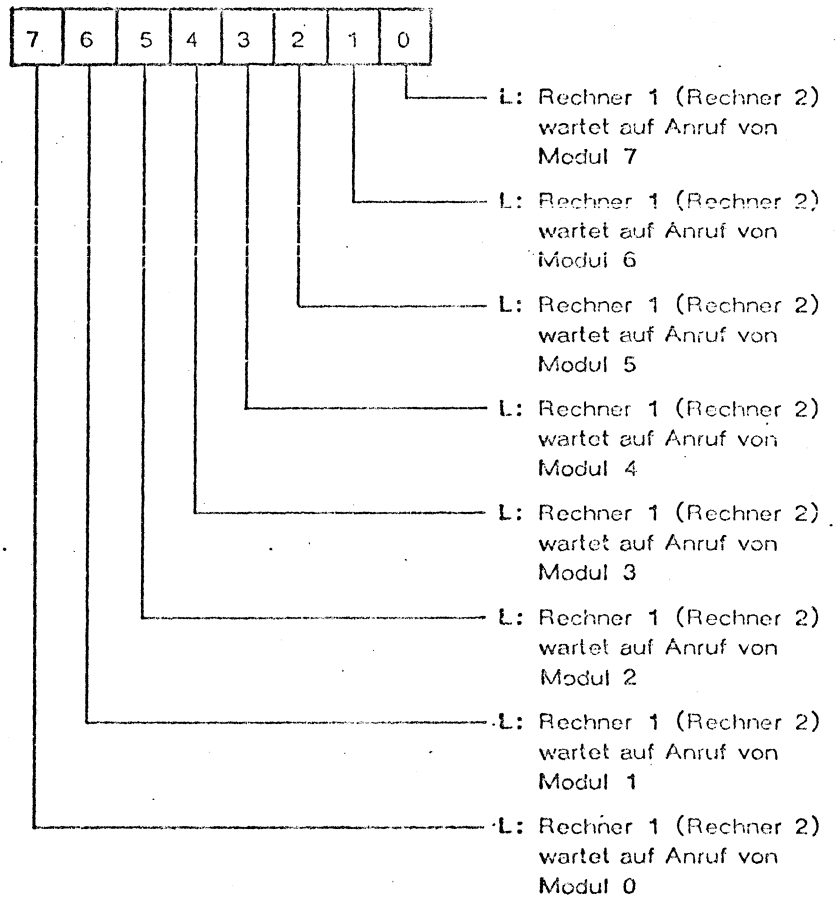
Datenspeicher Belegungsplan

Adressangabe oktal

DSP '26' Weichenbyte 2:



DSP '45' (DSP '46') Anrufbyte für Rechner 1 (bzw. Rechner 2):



DSP '51' Interruptbyte:

7	6	5	4	3	2	1	0
x	x	x	x	x	x	x	x

Interrupt von Modul 0 1 2 3 4 5 6 7 wurde bereits abgehandelt, d.h. wenn nötig und möglich ein Anruf gegeben.

DSP '211' AW-Adresse

7	6	5	4	3	2	1	0
x	x					x	x

AW-Adresse:

7	6	
0	0	AW-Nr. 0
0	L	AW-Nr. 1
L	0	AW-Nr. 2
L	L	AW-Nr. 3

Bit 1:
L: W32 BMP
0: W30 BMP

Bit 0:
L: HW-Koordinierung
0: keine HW-Koordinierung

1.1.3.
Vereinbarungen Merker:

MO: Wird gesetzt, wenn Wartungsmodus verlangt war.

MI: Wird von der Hardware gesetzt, wenn in MSL zuvor SMI gegeben wurde und die Indexmarke auf der Platte erkannt wird.

MU: Wird softwaremäßig gelöscht und nach 25 ms von der Hardware wieder gesetzt bzw. gesetzt bei Auftreten eines Fehlers während der Datenübertragung von und zum Laufwerk.

DPL: Kein Merker! Hiermit wird der Takt für das Laden von Diagnostik-Routinen simuliert (normalerweise wird der Takt von der Kassette gegeben). Zum Laden eines MIP-Wortes müssen zehn Bytes übernommen werden, daher auch zehnmal DPL gegeben werden. Der zeitliche Abstand zwischen zwei DPL's darf zwischen 2,2 µs und 130 µs liegen. Nach dem 10. DPL wird die Taktkette angehalten, bis das MIP-Wort geladen ist. Es ist danach ein Leerbefehl nötig, in dem aber wohl Bedingungen abgefragt werden dürfen. Vor dem DPL-Betrieb muß MC gesetzt werden. Tritt beim Laden des MIP-Wortes ein Fehler auf, so wird von der Hardware MC gelöscht

MC: a) Frei programmierbar

b) Wird gesetzt und mitverrechnet, wenn entsprechende Befehle gegeben werden (z.B. ADDP) und ein Überlauf errechnet wird.

MD: Wird hardwaremäßig gesetzt

a) beim Ende der Datenübertragung eines Bytes von und zum Laufwerk,

b) beim Auffinden des Snc-Bytes nachdem zuvor ST 4 (Bit 4 in ST) gegeben wurde,

c) bei Auffinden der AM nach AM-Search.

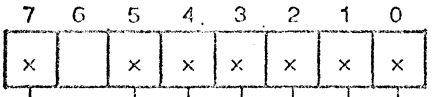
M5: Bei Sektorsuche softwaremäßig gesetzt, solange noch kein Sektor gefunden wurde

M6: Vom BMP gesetzt, wenn Formatierbefehl läuft (noch mehrere Verwendungen)

M7: Durch Setzen von M7 wird die automatische Datenphase angestoßen. Es muß gewährleistet sein, daß zuvor ZM ansteht. Ist die automatische Datenphase beendet, löscht die Hardware den M7.

Register:

LG: In LG befindet sich immer das 4. SIZ
HK: Weichenbyte 1



L: Anruf an R1 wurde noch nicht mit "Statusfrage nach Anruf" beantwortet

L: Anruf an R2 wurde noch nicht mit "Statusfrage nach Anruf" beantwortet

L: nach fehlerfreier Ausführung des Befehls AW freigeben

L: auf dieser Spur wurde bereits eine Adressenmarke gefunden (S, L, PL)

L: mindestens 1 gewünschten Sektor (S, L, PL) bereits gefunden

L: reserviert

L: adressiertes Laufwerk ist eine 200 MB-Version

1.1.4.
Konnektoren-Liste

Konnektor-Name	Definiert auf Adresse dez.	okt.	Blatt	angesprungen von Blatt
ADRF	1216	2300	250	
ALFE	1203	2263	270	260
	1248	2340		
AMWS	1488	2720	420	390,440,450
ANAB	2140	4134	170	160
ANGE	2089	4051	160	150
AWA	47	57	230	120
BEE	226	342	730	
BEES	678	1246	730	
BEE 1	213	325	710	390,450,570
BEFER	384	600	110	280,520,710
BEOD	252	374	280	290,330
BER	493	755	350	360,550,570
BERQ	1603	3103	570	
BERS 1	542	1036	350	360,380,520
BERX	521	1011	350	
BER 1	496	760	350	640
BER 2	497	761	350	590
BER 3	466	722	360	340
CEP	1518	2756	440	350
CORBE	1893	3545	640	620
COUN	776	1410	480	540
CTDA	1612	3114	630	610,620
CTDAF	1919	3577	640	630
CTLF	945	1661	400	370
CTLMZ	1252	2344	400	420
CTLR	509	775	400	340
DAFO	948	1664	500	
	1018	1772		
DAFO 1	868	1544	510	500
	1015	1767		
DAFO2	952	1670	540	510
	863	1537		
DALE	182	266	410	140,450,560
DAM	1512	2750	410	570
DASCH	190	276	560	140
DATA	1741	3315	610	
DBL	724	1324	370	410,440
DEF	1524	2764	440	
DICH	1477	2705	410	430
DICH 1	1481	2711	420	410,570
DIFF	2332	4434	310	300
DIFF 1	2355	4463	320	310
DL	1004	1754	410	440
DLF3	1501	2735	430	420
DLS	1516	2754	440	360
DPF	1084	2074	660	270
DPL	1188	2244	260	240
DPLN	1189	2245	260	270
DRE	341	525	230	250
DRLS	363	553	240	130
DRM	504	770	230	
DRS	1920	3600	180	180,250
DRV	1168	2220	250	240
DRW	1309	2435	240	
DS	1548	3014	570	440
DS 1	1560	3030	560	530
DS 1F	1600	3100	570	560

Konnektor-Name	Definiert auf Adresse dez.	okt.	Blatt	angesprungen von Blatt
ECCPX	962	1702	360	
ECC0	989	1735	390	380
ECC1	820	1464	350	
ECC2	821	1465	360	
ECOLI	1700	3244	580	
ERECC	1716	3264	600	
ERECO	1753	3331	620	
ERFI	1724	3274	600	
	1673	3211		
ERNOC	1699	3243	580	600,620
	1705	3251		
	1746	3322		
ERSBE	1085	2075	660	100,280,440
ERTS	445	675	750	420,510,550,570
	447	677		
ERTSQ	1318	2446	550	530,540
ERTSX	658	1222	510	460,470,480,490,500,540
ERTSY	1563	3033	570	560
FDL	940	1654	390	500,520,550,570,710
FDL1	941	1655	390	380,440,680
FDL2	1288	2410	710	430
FDR0	905	1611	500	370
FEA	1024	2000	650	140,390,410,620,700
FEA1	1083	2073	660	670
FEBE	1	1	100	660,740
FEBEX	716	1314	740	380,560
FEBE1	8	10	100	660
FEG	248	370	700	290,320,340,380,400,640,680,750
FEG1	249	371	700	290,340,530,630
FEG2	250	372	700	340,350,400
FEH1	1515	2753	410	440
FEH2	1513	2751	410	430
FEOR	1697	3241	600	580
FERT	2902	5526	780	770
FEUE	6	6	100	180
FEWE	1775	3357	620	610,630
FF1	1324	2454	540	530
	1328	2460		
FF2	1380	2544	550	540
	1377	2541		
	1312	2440		
FORF	1316	2444	530	550
	1438	2636		
FORM	990	1736	530	380
	991	1737		
FORM1	1292	2414	530	570
FUEB	2657	5451	330	390,610
GENB	2268	4334	140	700
GZM	1352	2510	540	530
IDRE	365	555	180	
IDRF	1101	2115	180	
IFE	372	564	340	380,420,460,630,760
ILNF	360	550	180	170
INFRA	2072	4030	150	
IRED	1831	3447	760	
KOFE	559	1057	740	470
KORF	2944	5600	610	600
KORFE	1751	3327	620	610

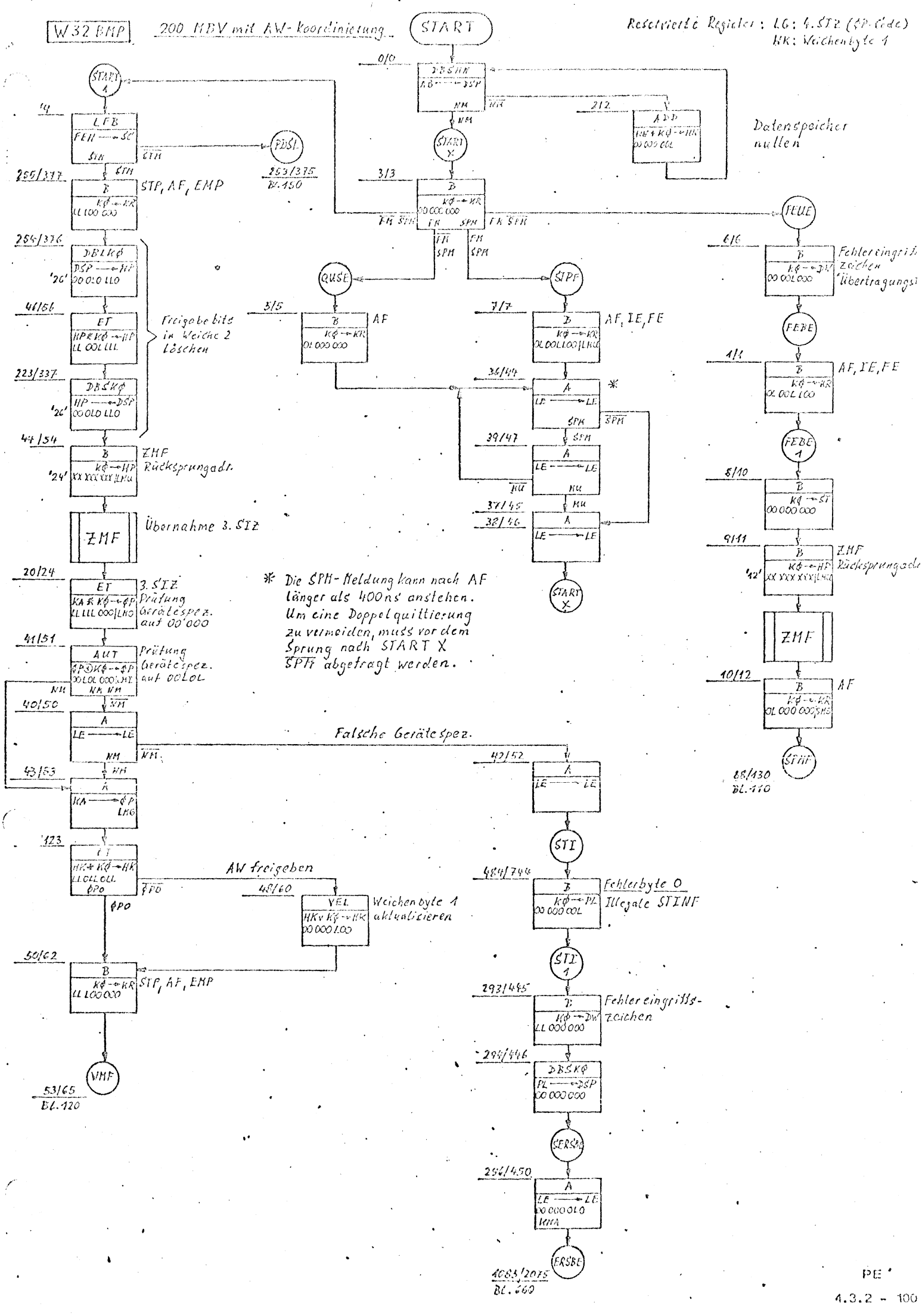
Konnektor-Name	Definiert auf Adresse dez.	okt.	Blatt	angesprungen von Blatt
LC6W	181	265	400	140
LEBE	637	1175	390	370,400,740
LESHA	420	644	340	400,530
LF1	499	763	350	340
LF2	532	1024	360	350
LF2Y	536	1030	360	400
LF3	586	1112	370	360
LF6	612	1144	380	370
LSPS	180	264	340	140
MOAR	2192	4220	190	210
MOSEF	90	132	700	280
MOSEL	2225	4261	200	190
MWR	483	743	340	420,460,760
MWR2	1306	2432	420	530
NAM	957 959	1675 1677	400	640
NAM1	961	1701	400	420
NEZE	1192	2250	260	
NOFOR	1409	2601	550	530
NORM	309	465	280	130
NOSEK	1553 1918	3021 3576	440	420
PDSL	253	375	150	100
PHYER	501	765	350	360,440
PLW	1003	1753	450	390
POSF	2844	5434	220	210
POSZ	2852	5444	330	320
PRL	1550	3016	440	
PRUEL	184	270	450	140
PSCOR	1653	3165	600	
QUSE	5	5	100	110,180
RECO	1744	3320	620	
REON	1816	3430	620	
SBW	141	215	230	
SDBE	904	1610	520	570
SCFO6	189	275	530	140
SCI6	188	274	460	140
SERSBE	296	450	100	130
SF1	704	1300	470	460
SF2	762	1372	480	470
SF3	816	1460	490	480
SF4	646	1206	500	490
SF5	907	1613	520	510
SNA	120	170	680	
SPMF	88	130	110	100
SPO	172	254	290	140
SPO1	2314	4412	300	290
SPRFEA	243	363	700	350
SPUF	1291	2413	710	540
SPUN	939	1653	520	
SPUN1	1281	2401	710	490
SPUN1X	717	1315	490	470,480,500
SSL	1296	2420	530	460
STAB	2229	4265	210	200
STAN	299	453	190	130
STARTS	860	1534	490	110,470,480,500,540,730

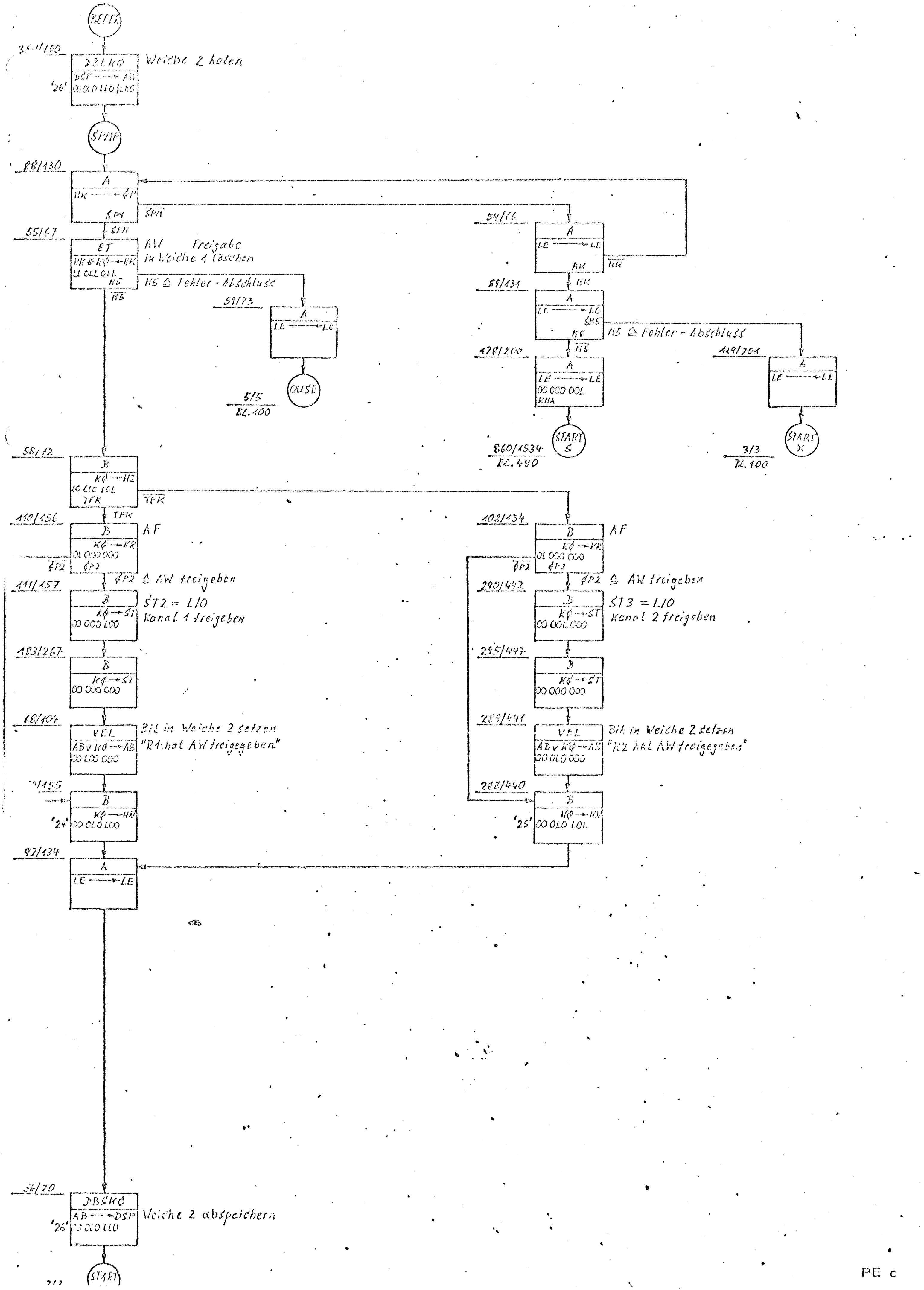
Konnektor-Name	Definiert auf Adresse dez.	okt.	Blatt	angesprungen von Blatt
STARTX	3	3	100	110,170,180,260,490,660,710
STARTXS	842	1512	490	730
STARTX 1	333 355	515 543	180	740
START 1	4	4	100	180
STATUE	2165	4165	140	130
STAWE	2278	4346	190	210
STFE	64	100	230	120
STI	484	744	100	120,130,240,290,300,410,680,710
STI 1	293	445	100	240
STPF	7	7	100	180
STST	1853	3475	230	120
STUE	2279	4347	230	210
STW	2248	4310	210	220
TERS	543	1037	380	360
TERO	1871	3517	640	
UKORF	2820	5404	590	580
UPSIZ	300	454	130	120
UPTAG	72	110	700	690
VERGL	965	1705	730	710,740
WMF	53	65	120	100
ZBIT	1207	2267	270	260

1.1.5.
Unterprogramm-Liste

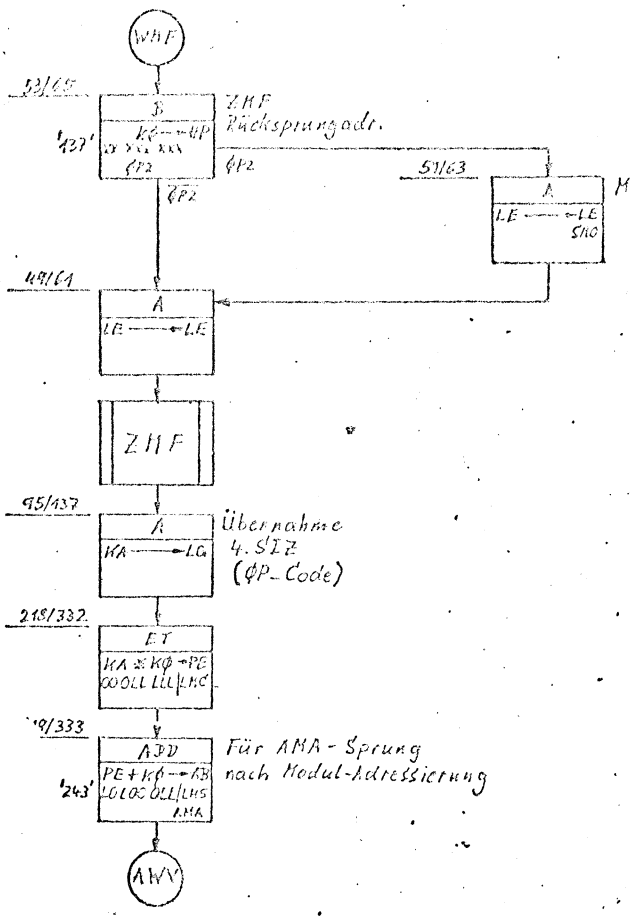
Unterprogramm	Definiert auf Blatt	angesprungen von Blatt
ADDUP	760	580,600
ADD 2	720	330,590,610
DRAS	720	220,330,390,590,610
ECCP	580	350,360,380
GAL	730	230,280,350,400,410,450
GALF	730	550
GALS	730	240,410,460,530
HCL	750	340,360
LESCA	680	340,400,410,460,530
MOBI	740	280,630
REST	760	620
REST 1	760	640
SHPU	760	600
TAG 2	720	190,210
ULAD	760	580
UPTAGF	710	190,200
UPTAGL	700	340
UPTAGL 1	700	420
UPTFE	710	240,250,650,660
USH 1	690	280,680
USH 2	690	290,340,680
USH 3	690	
USH 4	690	
USH 5	690	460
USH 6	690	410,430
USH 7	690	580
USH 8	690	630,760
USH 9	690	130,140,720
USH 10	690	290,320
USH 11	690	
USH 12	690	
ZGELB	770	390,590,610
ZMF	710	100,120,130
ZMFN	710	230,260,390,520,550
ZMFS	710	460
ZUEL	740	350,370
ZUEL 1	740	350,370,410

W32 BMP — 200 MBV mit AW-Koordinierung





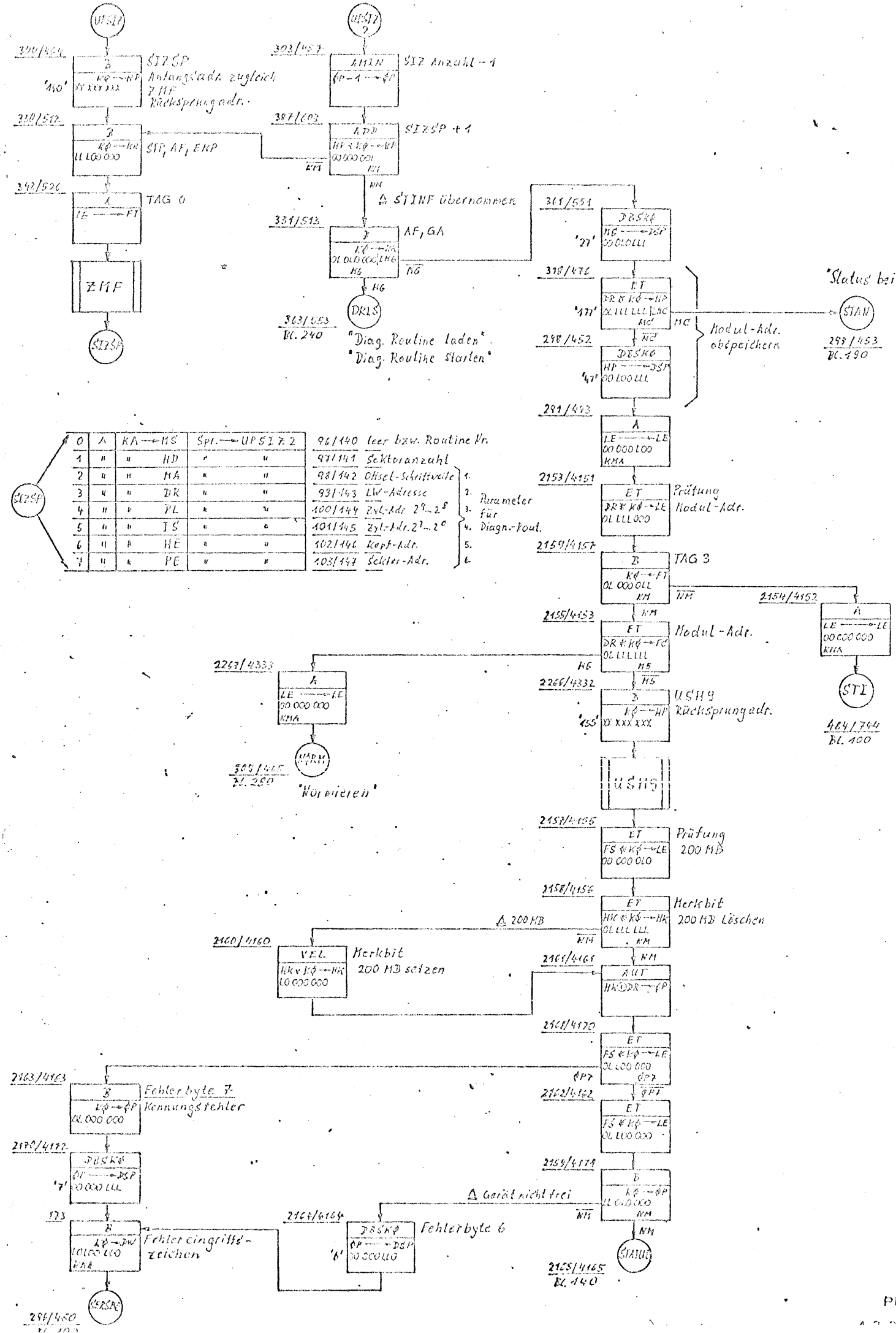
BEFEK
Weiche 2 holen
SPHF
AW Freigabe in Weiche 1 löschen
HS ≙ Fehler - Abschluss
CLOSE
5/5
BL.400
START S
860/1534
BL.490
START X
3/3
BL.400
AF
ØP2 ≙ AW freigeben
ST2 = L10
Kanal 1 freigeben
ST3 = L10
Kanal 2 freigeben
Bit in Weiche 2 setzen
"R1 hat AW freigegeben"
"R2 hat AW freigegeben"
Weiche 2 abspeichern
START



			*		**					
0	Illegale STINF	0						Spr.→STI		256/400
1	Status bei AN	1	B	KØ→ØP	00 000 00L | SHC	1 SIZ übernehmen	"	UPSIZ		257/401
2	" " Fehler	2	B	KØ→KR	0L 000 L00	AF, IE	"	STFE		258/402
3	" " Stat.Übert.	3	NB	KØ→KR	L0 LLL 0LL	AF, IE	"	STST (AMA)		259/403
4	Diagn.-Progr. Laden	4	B	KØ→ØP	00 00L 000 | SH6	6 SIZ übernehmen	"	UPSIZ		260/404
5	" " Starten	5	B	KØ→ØP	00 00L 000 | SH6	8 SIZ übernehmen	"	UPSIZ		261/405
6	Abholen AW Adresse	6	B	KØ→KR	0L 000 L00	AF, IE	"	AWA		262/406
7	Illegale STINF	7					"	STI		263/407
10	Modul Normieren	8	B	KØ→ØP	00 000 L00 | SH5	4 SIZ übernehmen	"	UPSIZ		264/410
11	Positionieren	9	B	KØ→ØP	00 000 LL0	6 SIZ übernehmen	"	UPSIZ		265/411
12	Lösche Reservierung	A	B	KØ→KR	0L 000 L00	AF, IE	"	LØRE (AMA)		266/412
13	Illegale STINF	B					"	STI		267/413
14	" "	C					"	STI		268/414
15	" "	D					"	STI		269/415
16	" "	E					"	STI		270/416
17	" "	F					"	STI		271/417
20	" "	10					"	STI		272/420
21	Sektor RO Lesen	11	B	KØ→ØP	00 000 LLL	7 SIZ übernehmen	"	UPSIZ		273/421
22	Count Lesen	12	B	KØ→ØP	00 000 LLL	7 SIZ übernehmen	"	UPSIZ		274/422
23	Daten Lesen	13	B	KØ→ØP	00 00L 000	8 SIZ übernehmen	"	UPSIZ		275/423
24	Illegale STINF	14					"	STI		276/424
25	Prüflesen	15	B	KØ→ØP	00 00L 000	8 SIZ übernehmen	"	UPSIZ		277/425
26	Illegale STINF	16					"	STI		278/426
27	" "	17					"	STI		279/427
30	" "	18					"	STI		280/430
31	Sektor RO Schreiben	19	B	KØ→ØP	00 000 LLL	7 SIZ übernehmen	"	UPSIZ		281/431
32	Formatieren	1A	B	KØ→ØP	00 000 LLL	7 SIZ übernehmen	"	UPSIZ		282/432
33	Daten Schreiben	1B	B	KØ→ØP	00 00L 000	8 SIZ übernehmen	"	UPSIZ		283/433
34	Illegale STINF	1C					"	STI		284/434
35	" "	1D					"	STI		285/435
36	" "	1E					"	STI		286/436
37	" "	1F					"	STI		287/437

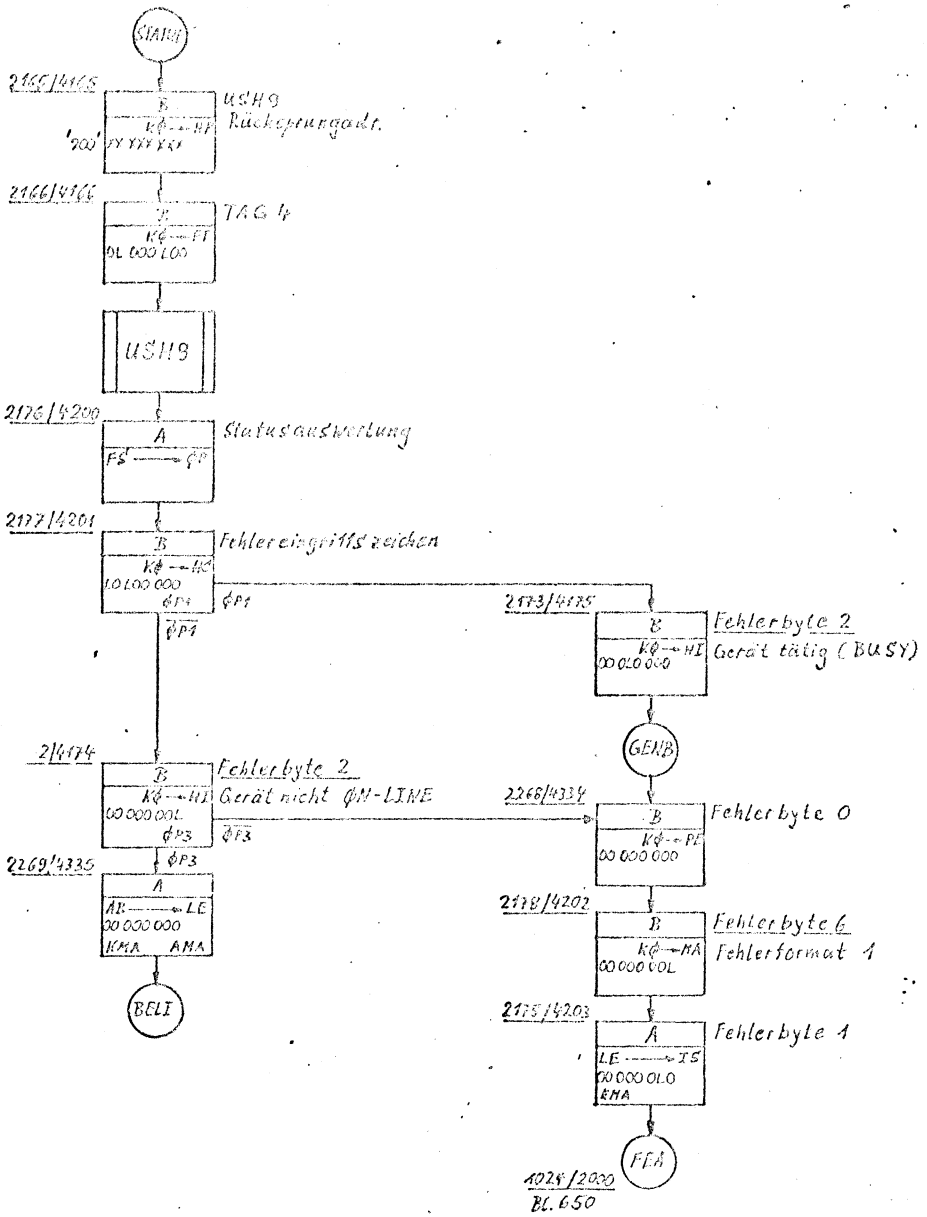
* Angaben sedezimal

** Konstante in ØP ist die Anzahl der noch zu übernehmenden SIZ



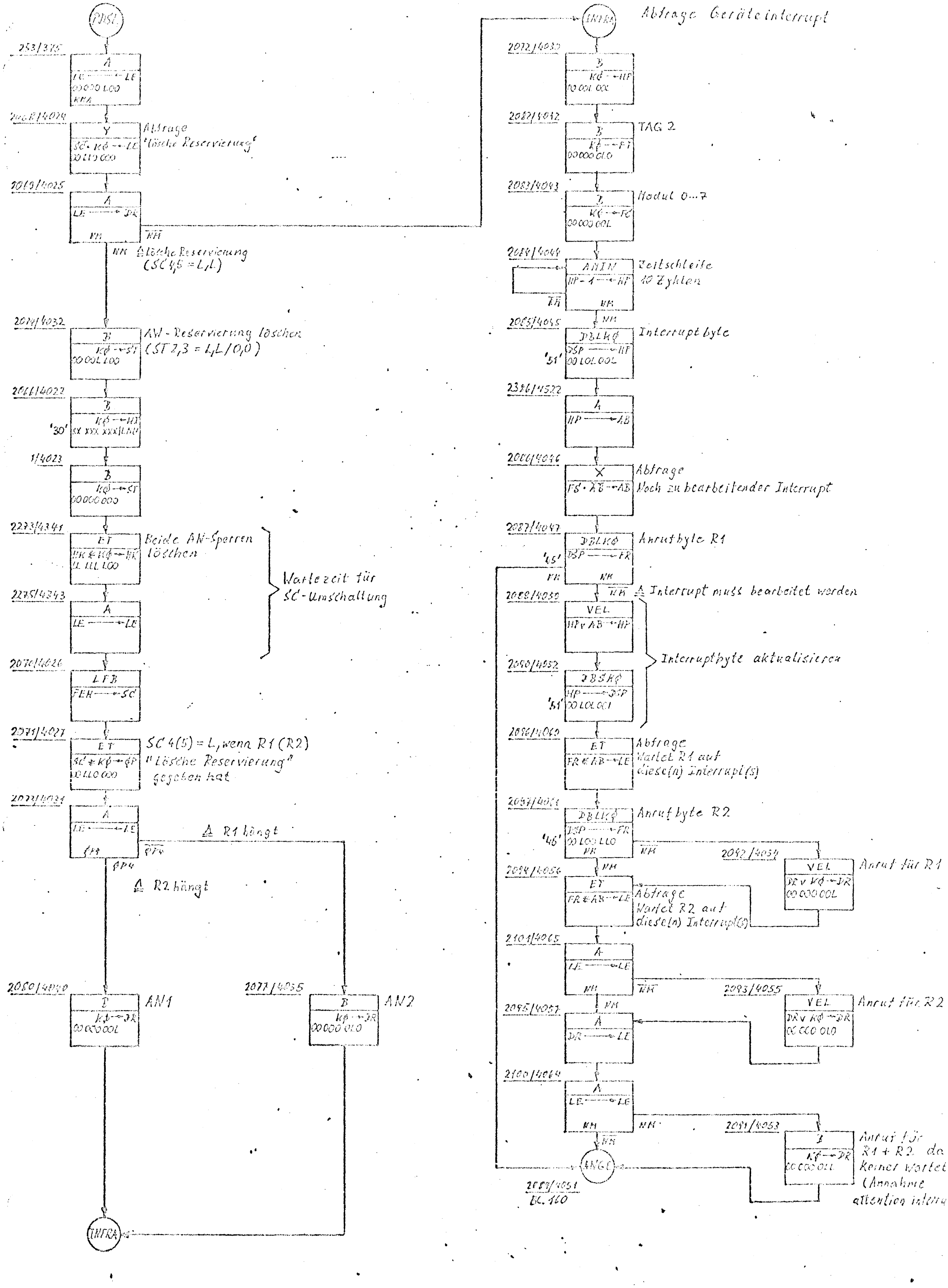
0	A	KA → MS	Spr. → UPSIZ 2	96/140	leer bzw. Routine Nr.	
1	"	" HD	" "	97/141	Sektoranzahl	
2	"	" MA	" "	98/142	Offset-Schrittweite	1.
3	"	" DR	" "	99/143	LW-Adresse	2.
4	"	" PL	" "	100/144	Zyl-Adr $2^9 \ldots 2^8$	3. Parameter für
5	"	" TS	" "	101/145	Zyl-Adr. $2^7 \ldots 2^0$	4. Diagn.-Rout.
6	"	" HE	" "	102/146	Kopf-Adr.	5.
7	"	" PE	" "	103/147	Sektor-Adr.	6.

PE

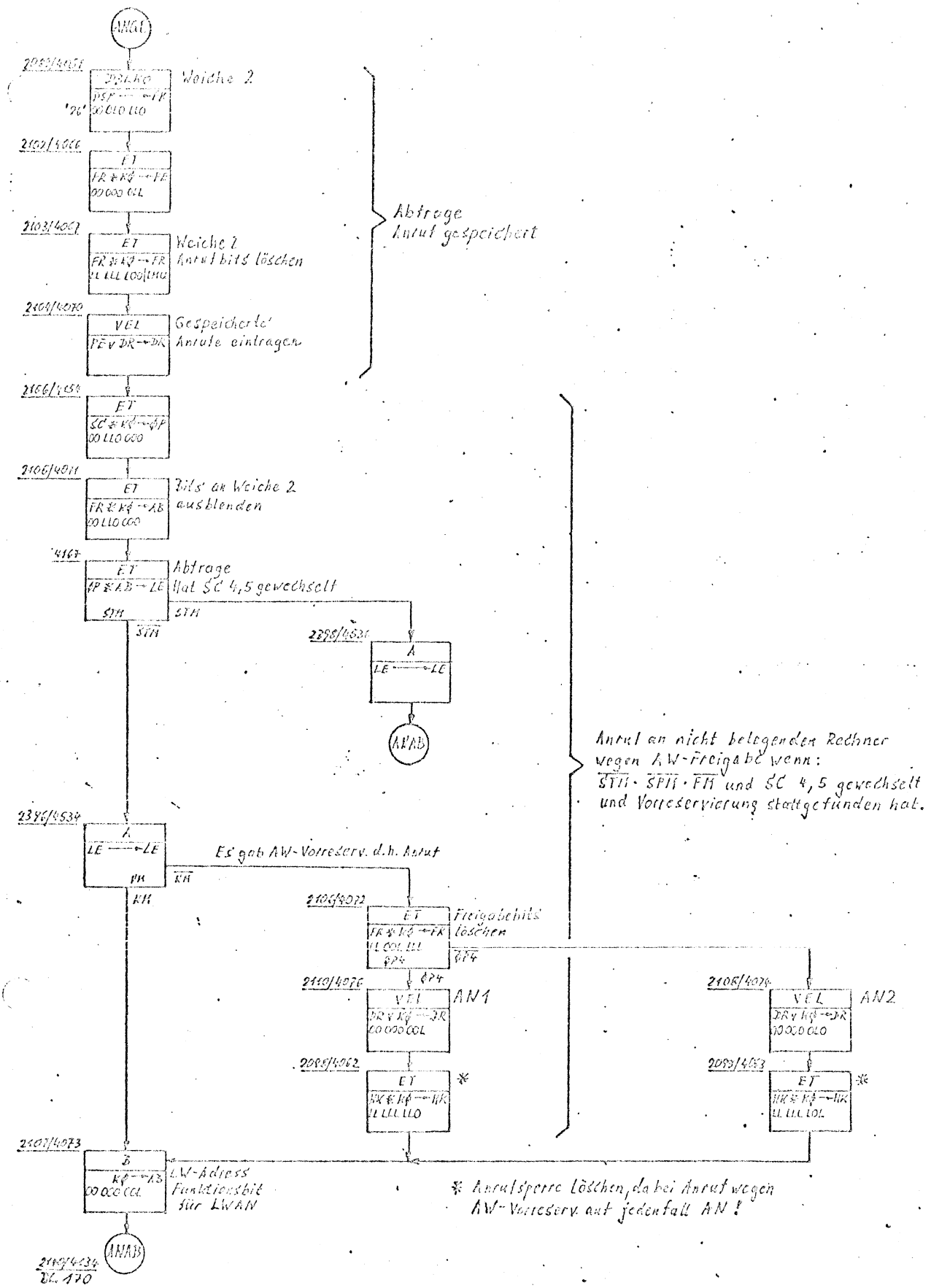


	Kan.Bef.	Anf.-Adr.	Blatt
Positionieren	SPΦ	172/254	210
Sektor RO Lesen	LSPS	180/264	270
Count Lesen	LCGW	181/265	340
Daten Lesen	DALE	182/266	350
Prüflesen	PRUEL	184/270	350
Sektor RO Schreiben	SCIG	188/274	410
Formatieren	SCIΦ6	189/275	480
Daten Schreiben	DASCH	190/276	350

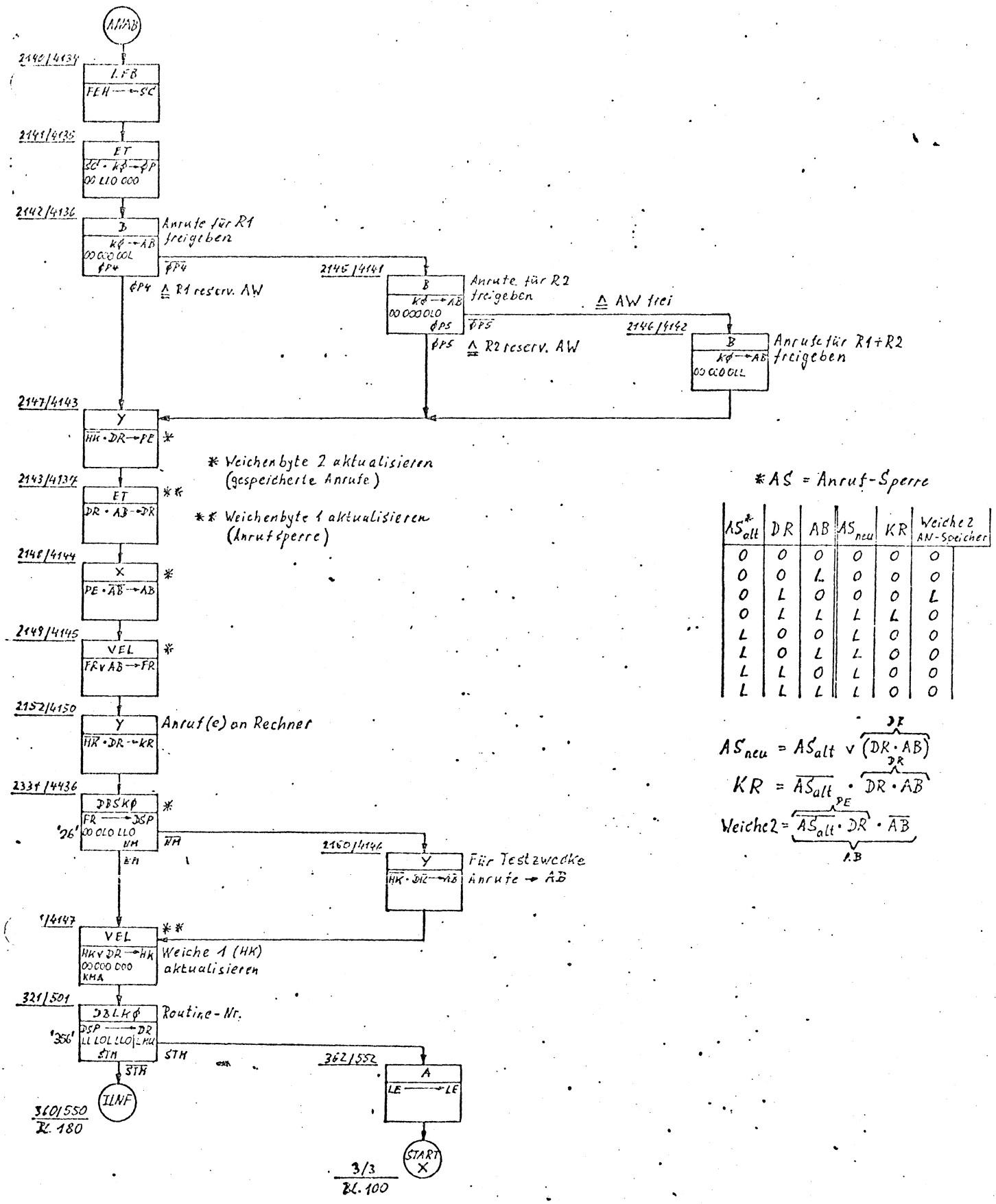
Die Liste BELI wird entsprechend ΦP-Code (in AB) per AMA angesprungen. Deshalb nicht zusammenhängender Bereich. Die (verbotenen) Werte dazwischen wurden schon vorher ausgeblendet. Die Befehle auf den jeweiligen Adressen siehe auf angegebenem Blatt.



PDSL
A
Abfrage
"Lösche Reservierung"
Δ Lösche Reservierung
(SC 4,5 = L,L)
AW-Reservierung löschen
(ST 2,3 = L,L/0,0)
'30'
Beide AN-Sperren
löschen
Wartezeit für
SC-Umschaltung
LFB
SC 4(5) = L, wenn R1 (R2)
"Lösche Reservierung"
gegeben hat
Δ R1 hängt
Δ R2 hängt
AN1
AN2
INTRA
Abfrage Geräteinterrupt
TAG 2
Modul 0...7
Zeitschleife
10 Zyklen
Interrupt byte
'51'
Abfrage
Noch zu bearbeitender Interrupt
Anrufbyte R1
'45'
Δ Interrupt muss bearbeitet werden
VEL
Interruptbyte aktualisieren
Abfrage
Wartet R1 auf
diese(n) Interrupt(s)
Anrufbyte R2
'46'
Anruf für R1
Abfrage
Wartet R2 auf
diese(n) Interrupt(s)
Anruf für R2
Anruf für
R1 + R2, da
keiner wartet
(Annahme
attention interrupt)
ANGC


ANRUF
2085/4011
'96'
Weiche 2
2102/4006
2103/4007
Weiche 1
Anruf bits löschen
2104/4010
Gespeicherte Anrufe eintragen
Abfrage
Anruf gespeichert
2106/4014
2106/4011
Bits an Weiche 2
ausblenden
4167
Abfrage
Hat SC 4,5 gewechselt
STH
2096/4024
ANAB
Anruf an nicht belegenden Rechner
wegen AW-Freigabe wenn:
STH · SPH · FH und SC 4,5 gewechselt
und Vorreservierung stattgefunden hat.
2336/4034
Es gab AW-Vorreserv. d.h. Anruf
KH
2109/4022
Freigabebits
löschen
2110/4026
AN1
2085/4062
2108/4024
AN2
2085/4063
2107/4023
LW-Adress
Funktionsbit
für LWAN
ANAB
2109/4034
DL 170
* Anrufsperre Löschen, da bei Anruf wegen
AW-Vorreserv. auf jeden fall AN!

PE



*AS = Anruf-Sperre

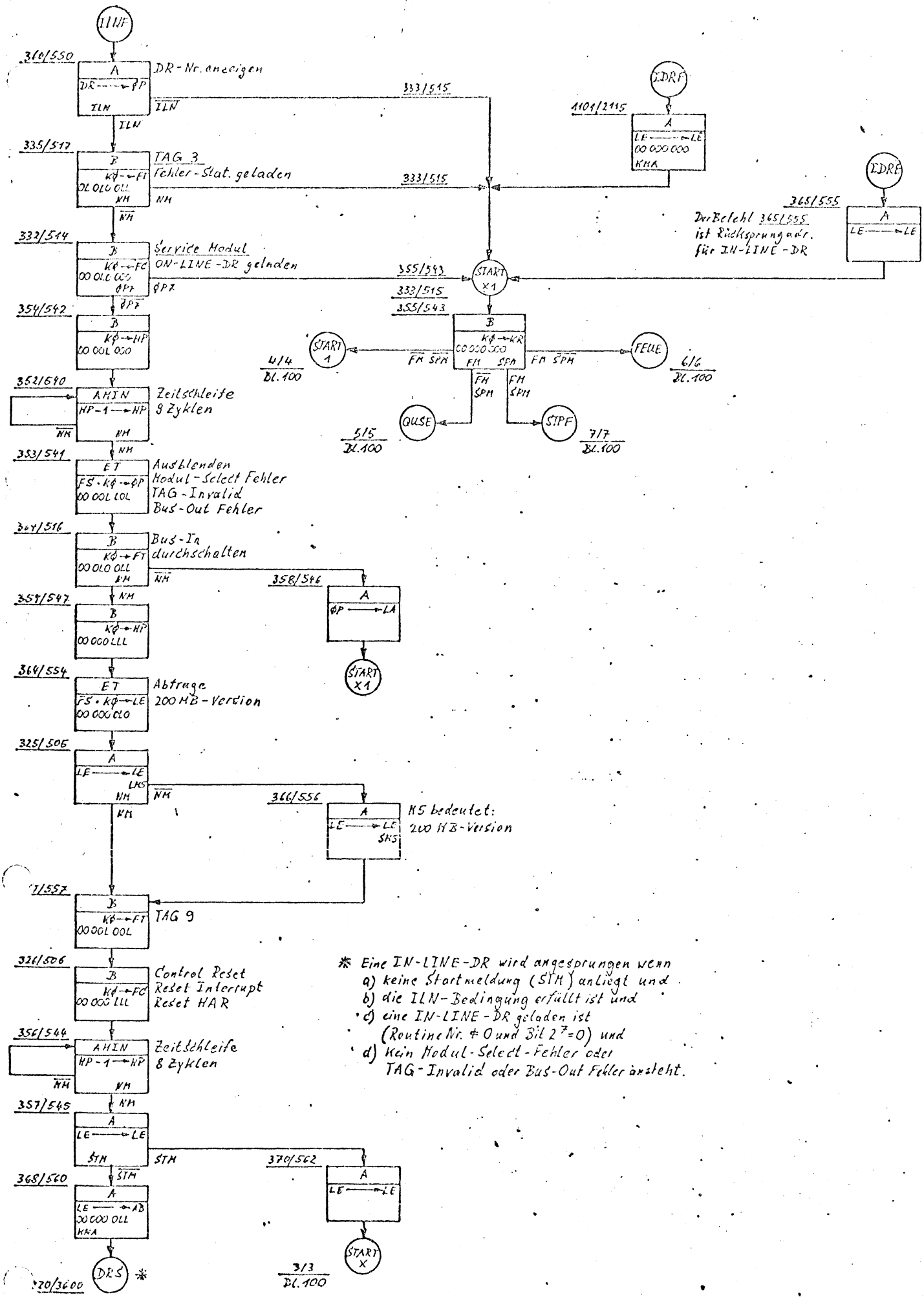
AS^*_{alt}	DR	AB	AS_{neu}	KR	Weiche 2 AN-Speicher
0	0	0	0	0	0
0	0	L	0	0	0
0	L	0	0	0	L
0	L	L	L	L	0
L	0	0	L	0	0
L	0	L	L	0	0
L	L	0	L	0	0
L	L	L	L	0	0

$$AS_{neu} = AS_{alt} \vee \overbrace{(DR \cdot AB)}^{DR}$$

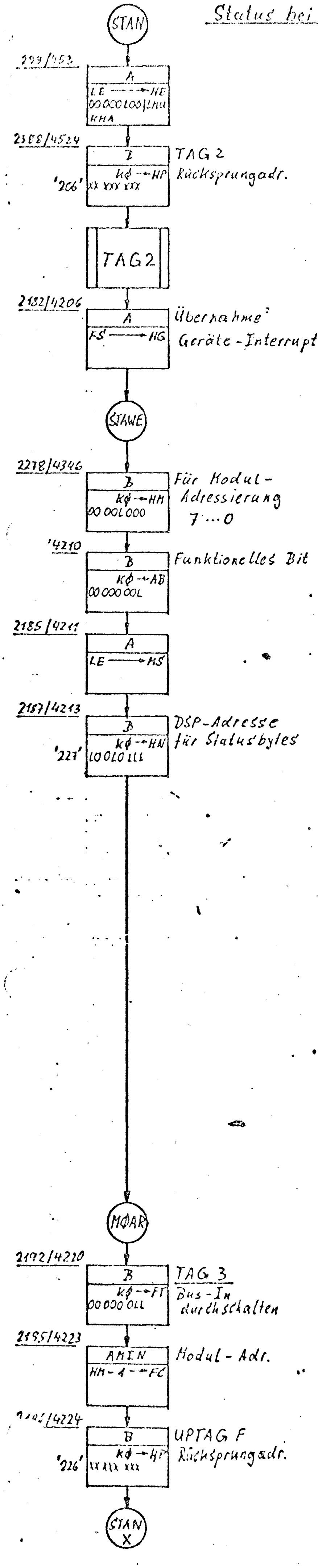
$$KR = \overline{AS_{alt}} \cdot \overbrace{DR \cdot AB}^{PE}$$

$$\text{Weiche2} = \underbrace{\overbrace{\overline{AS_{alt}} \cdot DR}^{PE} \cdot \overline{AB}}_{AB}$$

PE


ILNF
360/550
A
DR → ϕP
ILN
DR-Nr. anzeigen
333/515
ILN
IDRF
1101/2115
A
LE → LL
00 000 000
KKA
335/517
B
KϕFT
OL OLO OLL
NM
TAG 3
Fehler-Stat. geladen
333/515
NM
IDRE
365/555
A
LE → LE
Der Befehl 365/555 ist Rücksprungadr. für IN-LINE-DR
332/514
B
KϕFC
00 OLL 000
ϕPZ
Service Modul
ON-LINE-DR geladen
355/543
START X1
ϕPZ
332/515
355/543
354/542
B
KϕHP
00 00L 050
B
KϕKR
00 050 000
FM SPM
START 1
4/4 Bl. 100
FM SPM
FEHLE
6/6 Bl. 100
FM SPM
FM SPM
QUSE
5/5 Bl. 100
STPF
7/7 Bl. 100
352/690
AHIN
HP-1 → HP
NM
Zeitschleife 8 Zyklen
353/541
ET
FS · Kϕ → ϕP
00 00L L0L
Ausblenden
Modul-Select Fehler
TAG-Invalid
Bus-Out Fehler
364/516
B
KϕFT
00 0L0 0LL
NM
Bus-In durchschalten
358/546
A
ϕP → LA
START X1
359/547
B
KϕHP
00 000 LLL
364/554
ET
FS · Kϕ → LE
00 000 0L0
Abfrage 200 MB-Version
325/506
A
LE → LE
LKS
NM
366/556
A
LE → LE
SKS
KS bedeutet: 200 MB-Version
7/557
B
KϕFT
00 00L 00L
TAG 9
326/506
B
KϕFC
00 000 LLL
Control Reset
Reset Interrupt
Reset HAR
356/544
AHIN
HP-1 → HP
NM
Zeitschleife 8 Zyklen
357/545
A
LE → LE
STM
STM
370/562
A
LE → LE
368/560
A
LE → AB
00 000 0LL
KKA
DRS
*
START X
3/3 Bl. 100
* Eine IN-LINE-DR wird angesprungen wenn
a) keine Startmeldung (STM) anliegt und
b) die ILN-Bedingung erfüllt ist und
c) eine IN-LINE-DR geladen ist (Routine Nr. ≠ 0 und Bit 2^7 = 0) und
d) kein Modul-Select-Fehler oder TAG-Invalid oder Bus-Out Fehler ansteht.

Status bei Anruf

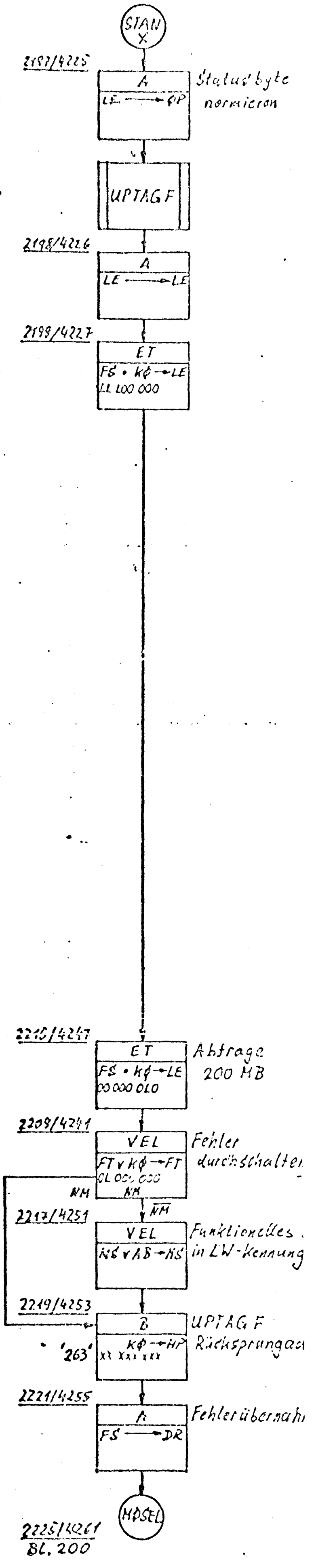


DSP-Belegung für STAN

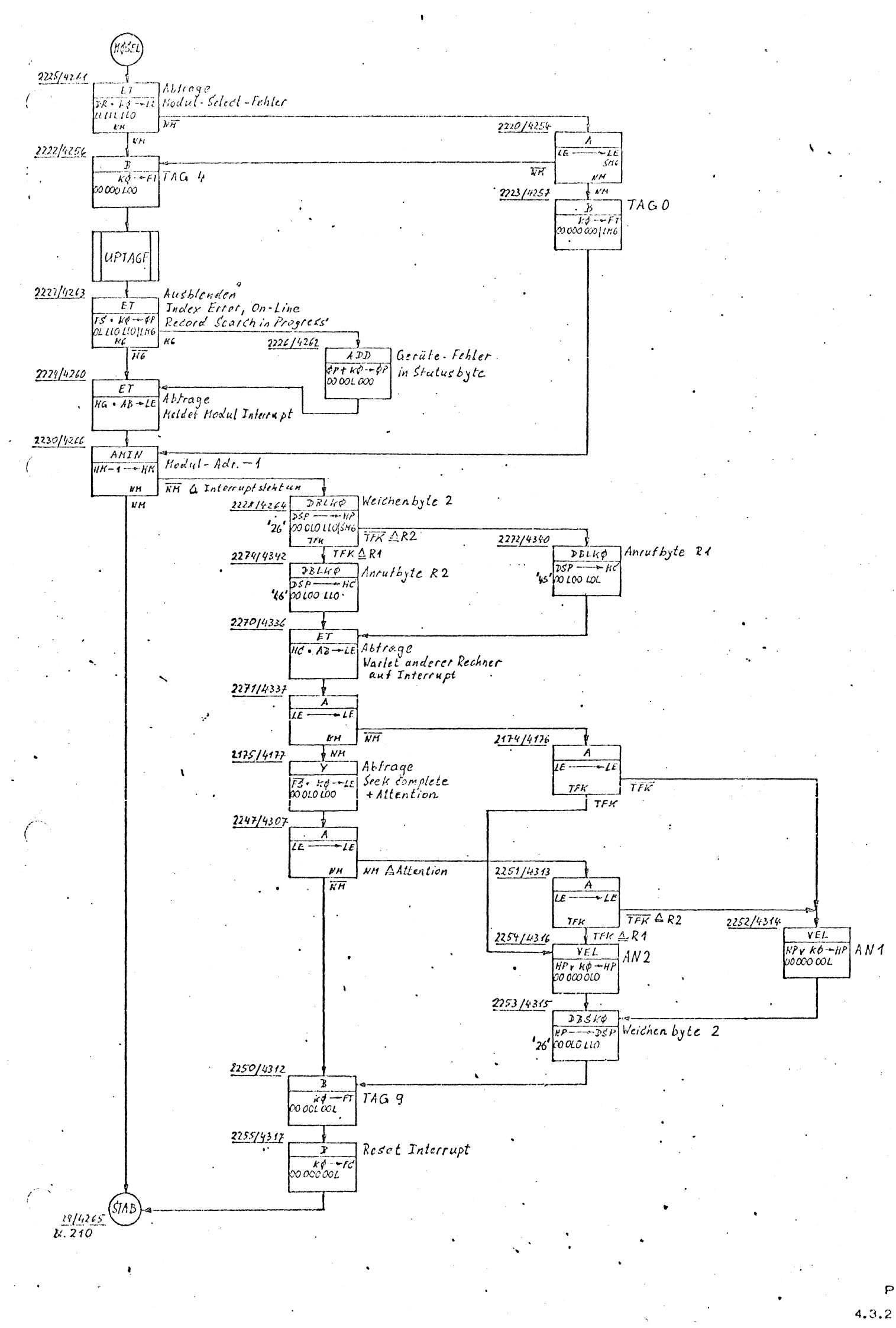
'220'	Modul 0	R1	'240'	Modul 0	R2
'221'	" 1		'241'	" 1	
'222'	" 2		'242'	" 2	
'223'	" 3		'243'	" 3	
'224'	" 4		'244'	" 4	
'225'	" 5		'245'	" 5	
'226'	" 6		'246'	" 6	
'227'	" 7		'247'	" 7	
'230'	3/6 Adr. bei *	HE	'250'	3/6 Adr. bei *	HE
'231'	LW-Kennung	MS	'251'	LW-Kennung	MS

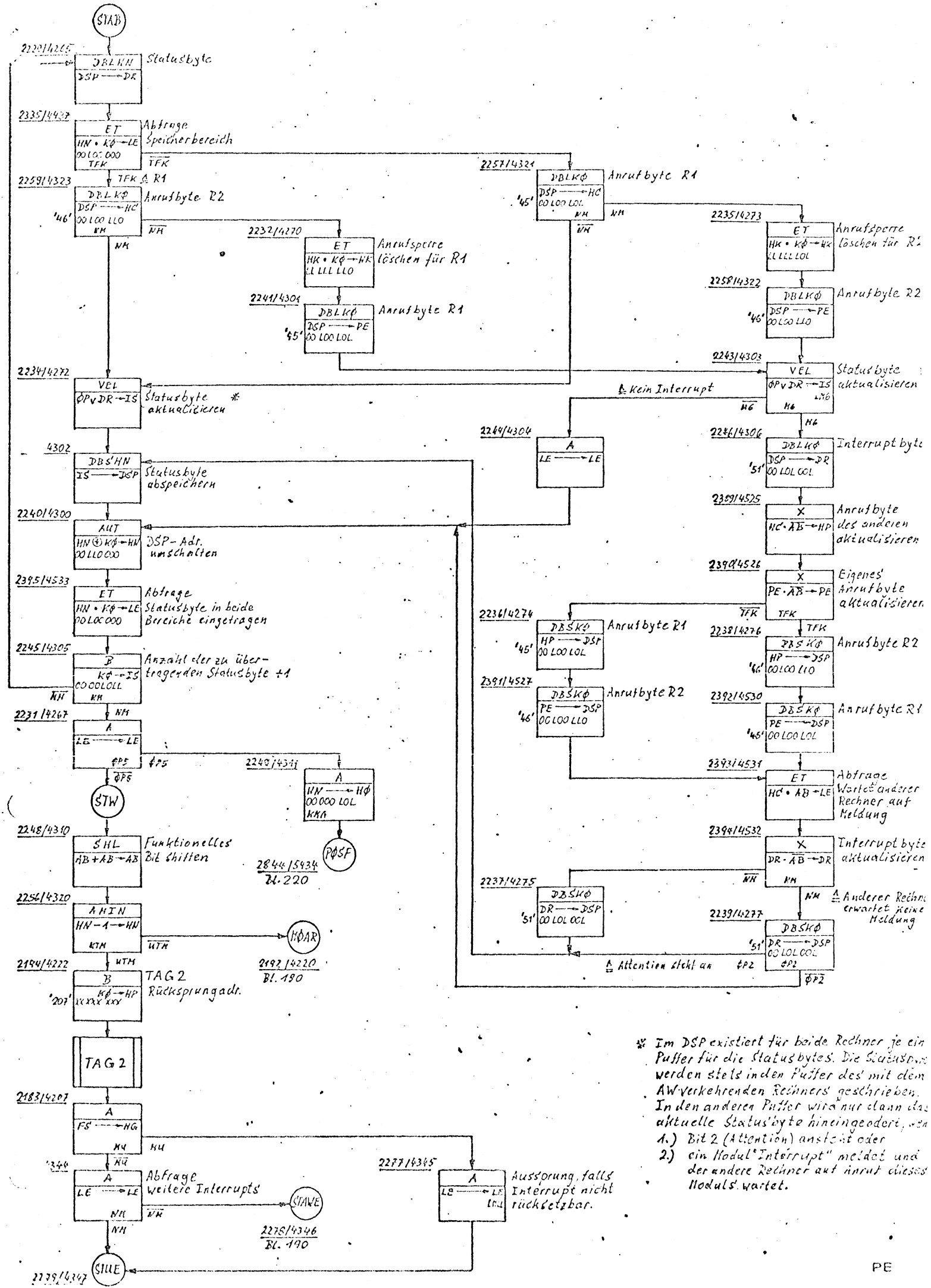
* Pos.-Fehler-Überlauf

Belegte Register bei Einsprung: LG: 4. SIZ (φP-Code)
HK: Weichenbyte 1



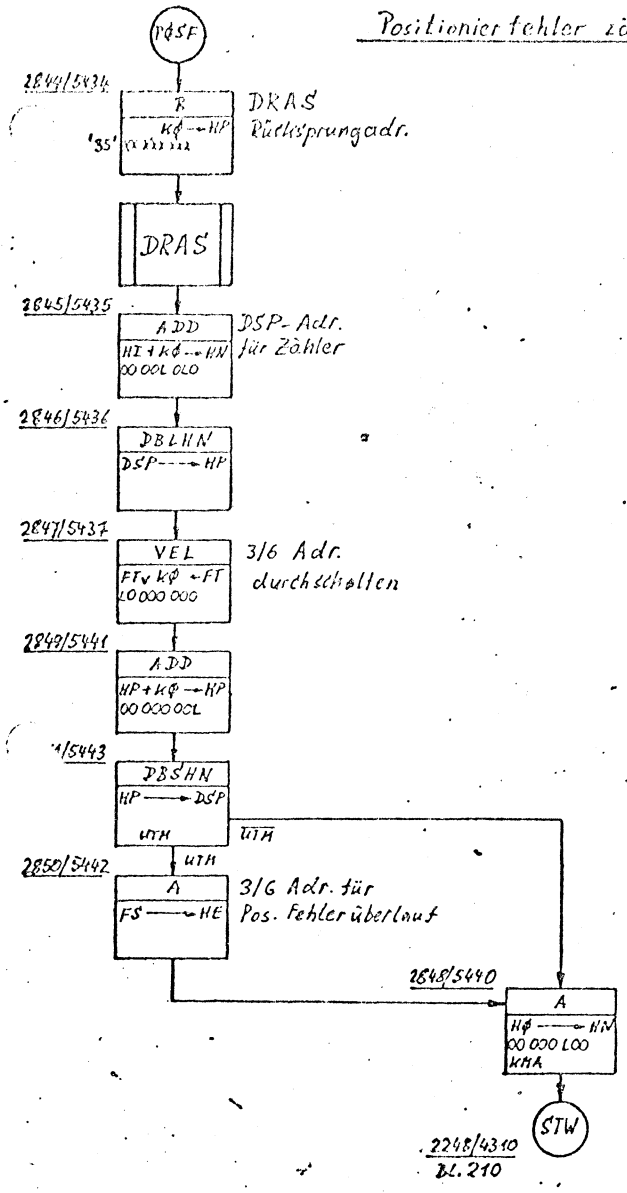
PE


MASEL
2225/4241
ET
Abfrage
Modul-Select-Fehler
2220/4254
A
2222/4256
B
TAG 4
2223/4257
B
TAG 0
UPTAGE
2227/4263
ET
Ausblenden
Index Error, On-Line
Record Search in Progress
2226/4262
ADD
Geräte-Fehler
in Statusbyte
2229/4260
ET
Abfrage
Meldet Modul Interrupt
2230/4266
AMIN
Modul-Adr. −1
NM ≙ Interrupt steht an
2228/4264
'26'
DBLKΦ
Weichenbyte 2
TFK ≙ R2
2272/4340
DBLKΦ
Anrufbyte R1
'45'
2274/4342
TFK ≙ R1
DBLKΦ
Anrufbyte R2
'46'
2270/4336
ET
Abfrage
Wartet anderer Rechner
auf Interrupt
2271/4337
A
2174/4176
A
2175/4177
Y
Abfrage
Seek complete
+ Attention
2247/4307
A
NM ≙ Attention
2251/4313
A
TFK ≙ R2
2252/4314
VEL
AN1
2254/4316
TFK ≙ R1
VEL
AN2
2253/4315
DBSKΦ
Weichenbyte 2
'26'
2250/4312
B
TAG 9
2255/4317
X
Reset Interrupt
STAB
19/4265
K. 210


STAB
Statusbyte
Abfrage Speicherbereich
TFK ≙ R1
Anrufbyte R2
Anrufsperre löschen für R1
Anrufbyte R1
Statusbyte aktualisieren *
Statusbyte abspeichern
DSP-Adr. umschalten
Abfrage Statusbyte in beide Bereiche eingetragen
Anzahl der zu übertragenden Statusbyte +1
STW
Funktionelles Bit shiften
TAG 2
Rückspringadr.
Abfrage weitere Interrupts
SHVE
SHLE
Anruf byte R1
Anruf byte R2
Anrufsperre löschen für R2
Interrupt byte
Anrufbyte des anderen aktualisieren
Eigenes Anrufbyte aktualisieren
Abfrage Warte auf anderen Rechner auf Meldung
≙ kein Interrupt
≙ Attention steht an
≙ Anderer Rechner erwartet keine Meldung
Aussprung, falls Interrupt nicht rücksetzbar.
* Im DSP existiert für beide Rechner je ein Puffer für die Statusbytes. Die Statusbytes werden stets in den Puffer des mit dem AW verkehrenden Rechners geschrieben. In den anderen Puffer wird nur dann das aktuelle Statusbyte hineingeodert, wenn
1.) Bit 2 (Attention) ansteht oder
2.) ein Modul "Interrupt" meldet und der andere Rechner auf Anruf dieses Moduls wartet.

PE

Positionier fehler zählen

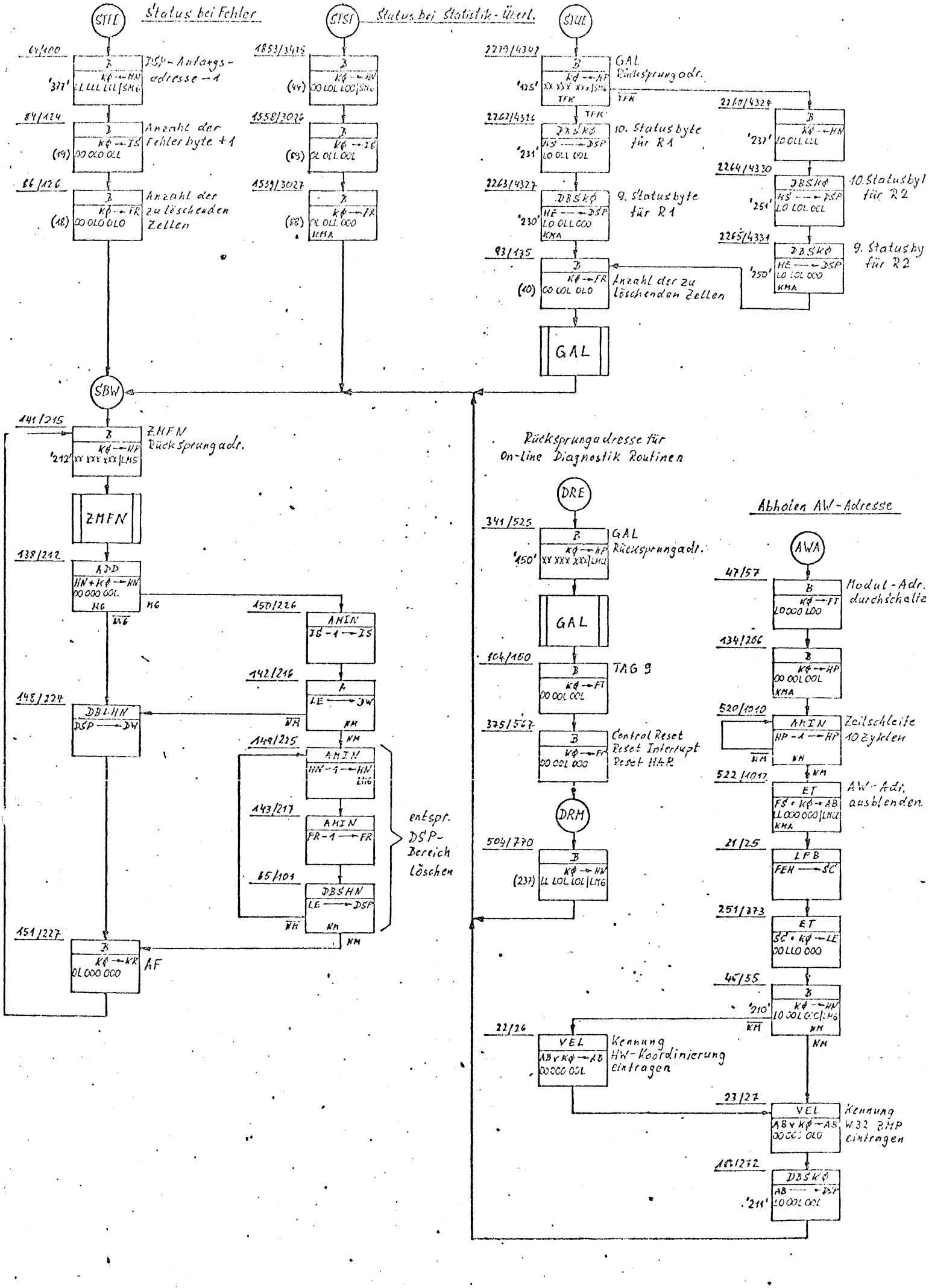


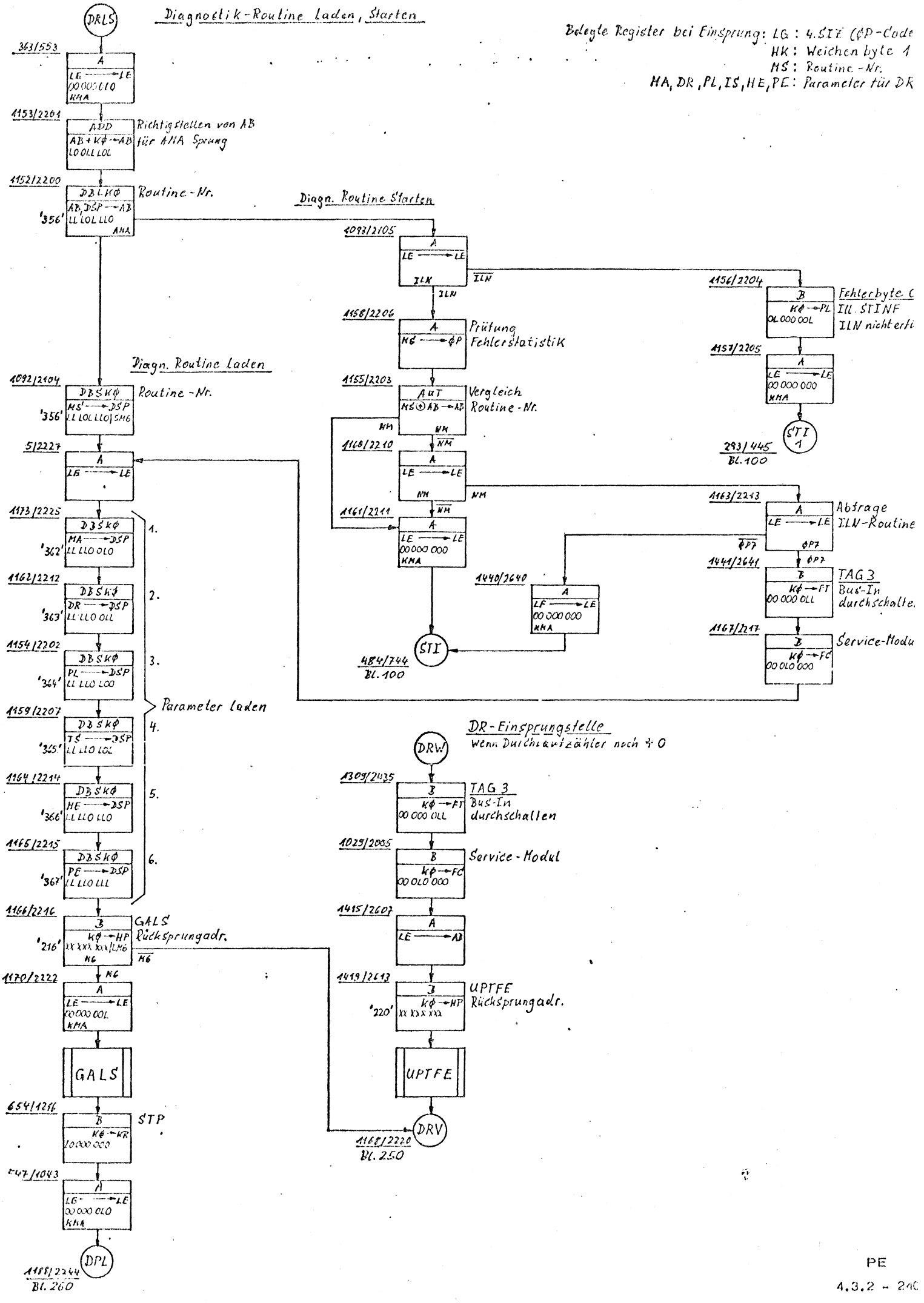
DSP-Belegung für Fehlerstatistik (Modul A)

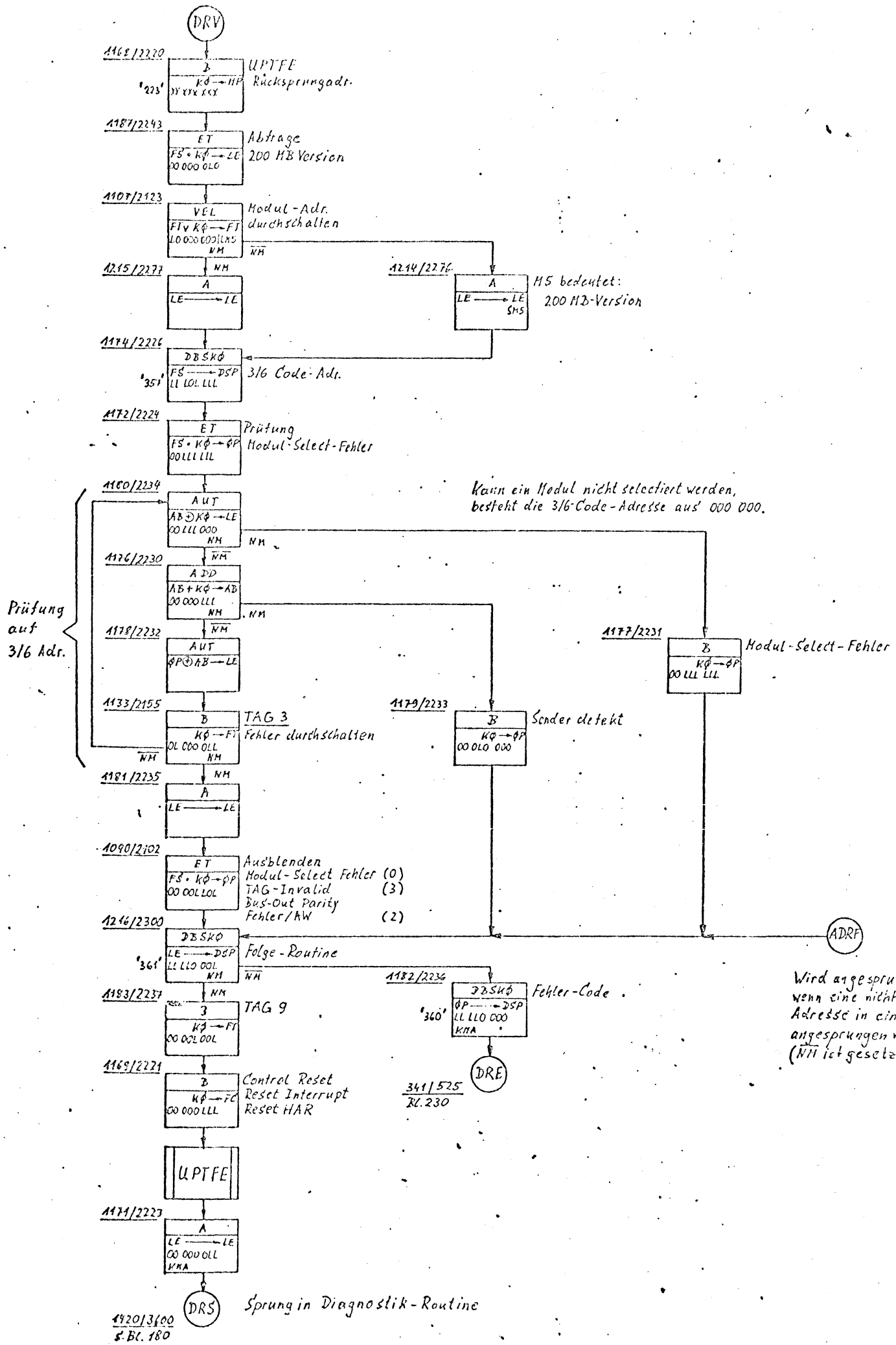
'55'		Anzahl der Gelés. Bytes (max. 8 589 934 591)
'56'		
'57'		
'60'		
'61'		Korrigierbare Lesefehler (max. 65 535)
'62'		
'63'		Unkorrigierbare Lesefehler (max. 65 535)
'64'		
'65'		Positionier vorgänge (max. 65 535)
'66'		
'67'		Positionier fehler (max. 255)

Statusbyte

0	Laufwerk für anderen Zugreifer reserv.
1	Laufwerk arbeitet
2	Attention
3	Geräte fehler
4	Positionieren beendet
5	Positionierfehler
6	Offset aktiv
7	AW für anderen Zugreifer reserv. ($2^6 \ldots 2^0 = 0$)

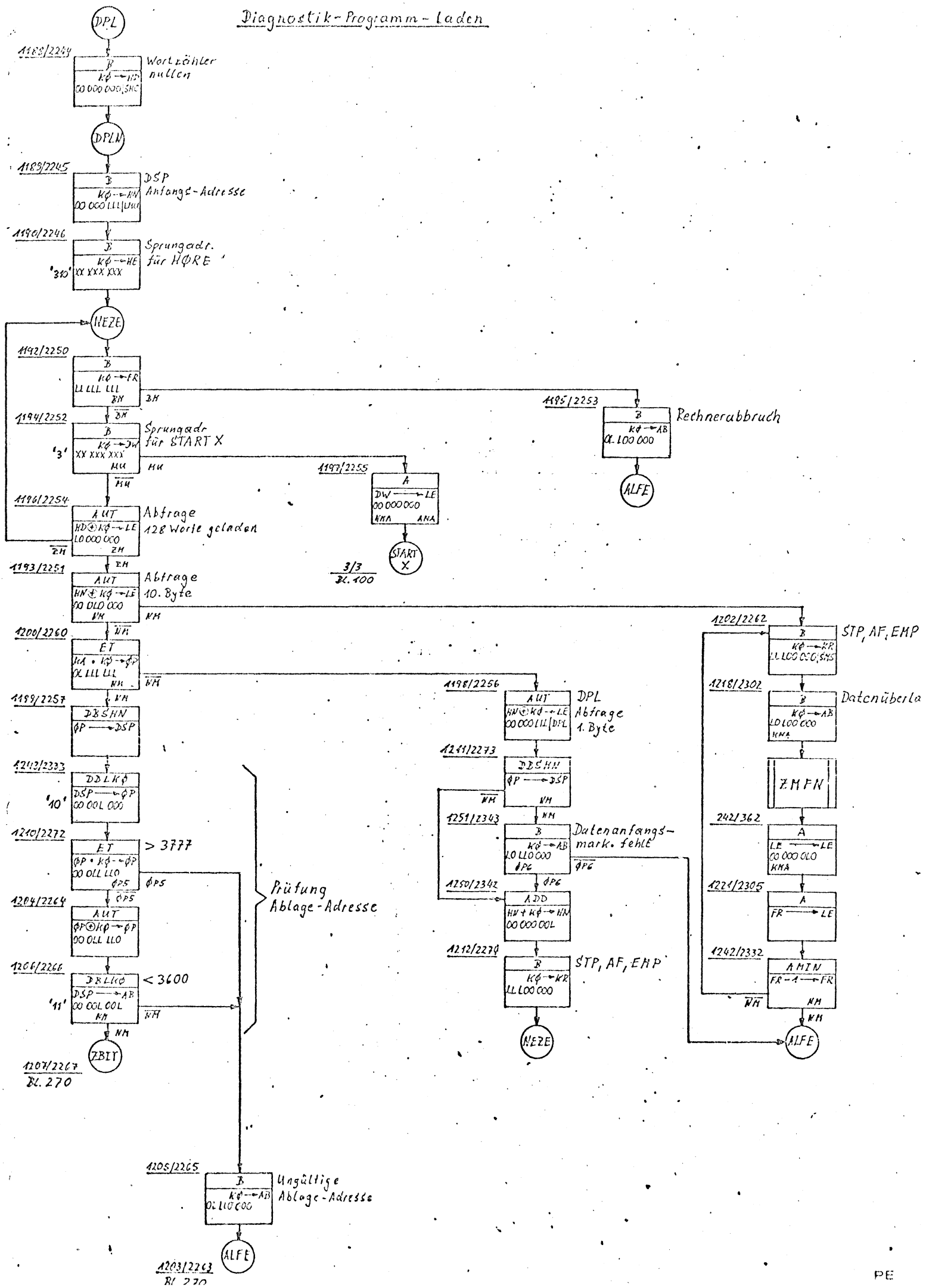

STFE
Status bei Fehler
STSI
Status bei Statistik-Übert.
STUE
GAL Rücksprungadr.
DSP-Anfangs-adresse-1
Anzahl der Fehlerbyte +1
Anzahl der zu löschenden Zellen
10. Statusbyte für R1
9. Statusbyte für R1
10. Statusbyt für R2
9. Statusby für R2
Anzahl der zu löschenden Zellen
GAL
SBW
ZMFN Rücksprungadr.
ZMFN
Rücksprungadresse für On-Line Diagnostik Routinen
DRE
GAL Rücksprungadr.
GAL
TAG 9
Control Reset Reset Interrupt Reset HAR
DRM
Abholen AW-Adresse
AWA
Modul-Adr. durchschalte
Zeilschleife 10 Zyklen
AW-Adr. ausblenden
entspr. DSP-Bereich löschen
AF
Kennung HW-Koordinierung eintragen
Kennung W32 BMP eintragen


Diagnostik-Routine Laden, Starten
Belegte Register bei Einsprung: LG : 4.STF (ΦP-Code
HK : Weichen byte 1
MS : Routine-Nr.
MA, DR, PL, IS, HE, PE : Parameter für DR
DRLS
363/553
A
LE → LE
00 000 110
KMA
1153/2261
ADD
AB + KΦ → AB
LO OLL LOL
Richtigstellen von AB für AMA Sprung
1152/2200
DBS KΦ
AB, DSP → AB
LL LOL LLO
'356'
AMA
Routine-Nr.
Diagn. Routine Starten
4093/2105
A
LE → LE
ILN
ILN
1156/2204
B
KΦ → PL
OL 000 00L
Fehlerbyte C
III. STINF
ILN nicht erfü
1157/2205
A
LE → LE
00 000 000
KMA
STI 1
293/445
Bl. 100
1158/2206
A
KG → ΦP
Prüfung Fehlerstatistik
1155/2203
AUT
MS ⊕ AB → AP
Vergleich Routine-Nr.
NM
NM
1160/2210
A
LE → LE
NM
NM
1161/2211
A
LE → LE
00 000 000
KMA
1163/2213
A
LE → LE
ΦP7
Abfrage ILN-Routine
ΦP7
1441/2641
B
KΦ → FT
00 000 0LL
TAG 3 Bus-In durchschalte
1167/2217
B
KΦ → FC
00 0L0 000
Service-Modu
1440/2640
A
LE → LE
00 000 000
KMA
SII
484/744
Bl. 100
Diagn. Routine Laden
1092/2104
DBS KΦ
MS' → DSP
LL LOL LLO SM6
'356'
Routine-Nr.
5/2227
A
LE → LE
1173/2225
DBS KΦ
MA → DSP
LL LLO 0L0
'362'
1.
1162/2212
DBS KΦ
DR → DSP
LL LLO 0LL
'363'
2.
1154/2202
DBS KΦ
PL → DSP
LL LLO L00
'364'
3.
Parameter laden
1159/2207
DBS KΦ
IS → DSP
LL LLO L0L
'365'
4.
1164/2214
DBS KΦ
HE → DSP
LL LLO LL0
'366'
5.
1166/2215
DBS KΦ
PE → DSP
LL LLO LLL
'367'
6.
1166/2216
B
KΦ → HP
XX XXX XXX/LM6
HG
'216'
GALS Rücksprungadr.
HG
1170/2222
A
LE → LE
00 000 00L
KMA
GALS
654/1214
B
KΦ → KR
00 000 000
STP
647/1043
A
LG → LE
00 000 0L0
KMA
DPL
1188/2244
Bl. 260
DR-Einsprungstelle
Wenn Durchlaufzähler noch ≠ 0
DRV
1309/2435
B
KΦ → FT
00 000 0LL
TAG 3 Bus-In durchschalten
1029/2005
B
KΦ → FC
00 0L0 000
Service-Modul
1415/2607
A
LE → AB
1419/2613
B
KΦ → HP
'220'
XX XXX XXX
UPTFE Rücksprungadr.
UPTFE
DRV
1168/2220
Bl. 250

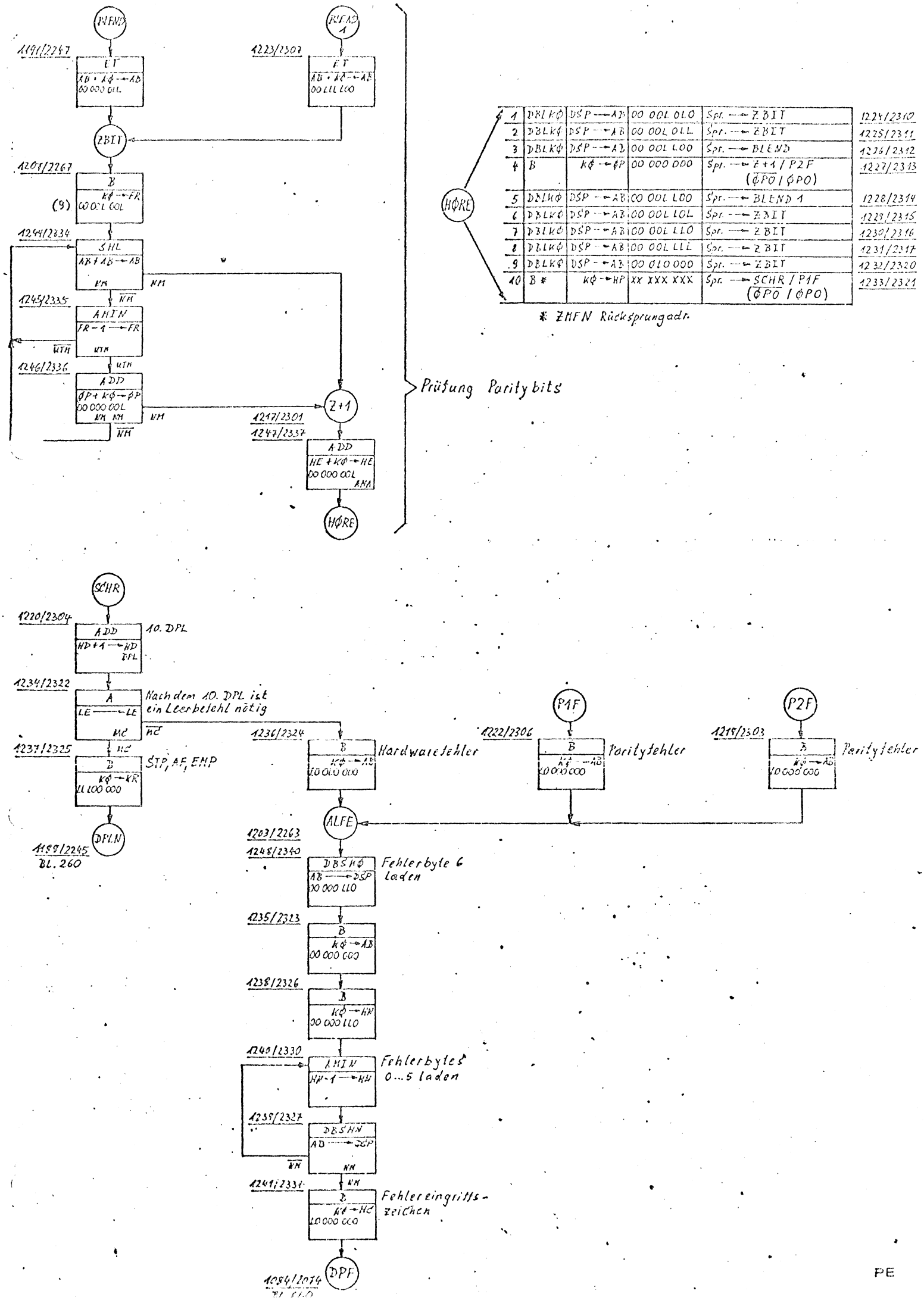

DRV
1168/2220
B
KØ → HP
'223'
UPTFE Rücksprungadr.
1187/2243
ET
FS • KØ → LE
Abfrage 200 MB Version
1107/2123
VEL
FI v KØ → FI
Modul-Adr. durchschalten
NM
1215/2277
A
LE → LE
1214/2276
A
LE → LE SMS
MS bedeutet: 200 MB-Version
1174/2226
DB SKØ
FS → DSP
'351'
3/6 Code-Adr.
1172/2224
ET
FS • KØ → ØP
Prüfung Modul-Select-Fehler
1180/2234
AUT
AB ⊕ KØ → LE
Kann ein Modul nicht selectiert werden, besteht die 3/6-Code-Adresse aus 000 000.
1176/2230
ADD
AB + KØ → AB
1178/2232
AUT
ØP ⊕ AB → LE
Prüfung auf 3/6 Adr.
1133/2155
B
KØ → FI
TAG 3 Fehler durchschalten
1179/2233
B
KØ → ØP
Sender defekt
1177/2231
B
KØ → ØP
Modul-Select-Fehler
1181/2235
A
LE → LE
1090/2102
ET
FS • KØ → ØP
Ausblenden Modul-Select Fehler (0) TAG-Invalid (3) Bus-Out Parity Fehler/AW (2)
1246/2300
DB SKØ
LE → DSP
'361'
Folge-Routine
1182/2236
DB SKØ
ØP → DSP
'360'
Fehler-Code
341/525
Bl. 230
DRE
ADRF
Wird angesprungen, wenn eine nicht belegte Adresse in einer DR angesprungen wird. (NM ist gesetzt)
1183/2237
3
KØ → FI
TAG 9
1169/2221
B
KØ → FC
Control Reset Reset Interrupt Reset HAR
UPTFE
1171/2223
A
LE → LE
DRS
Sprung in Diagnostik-Routine
1920/3400
s. Bl. 180

PE

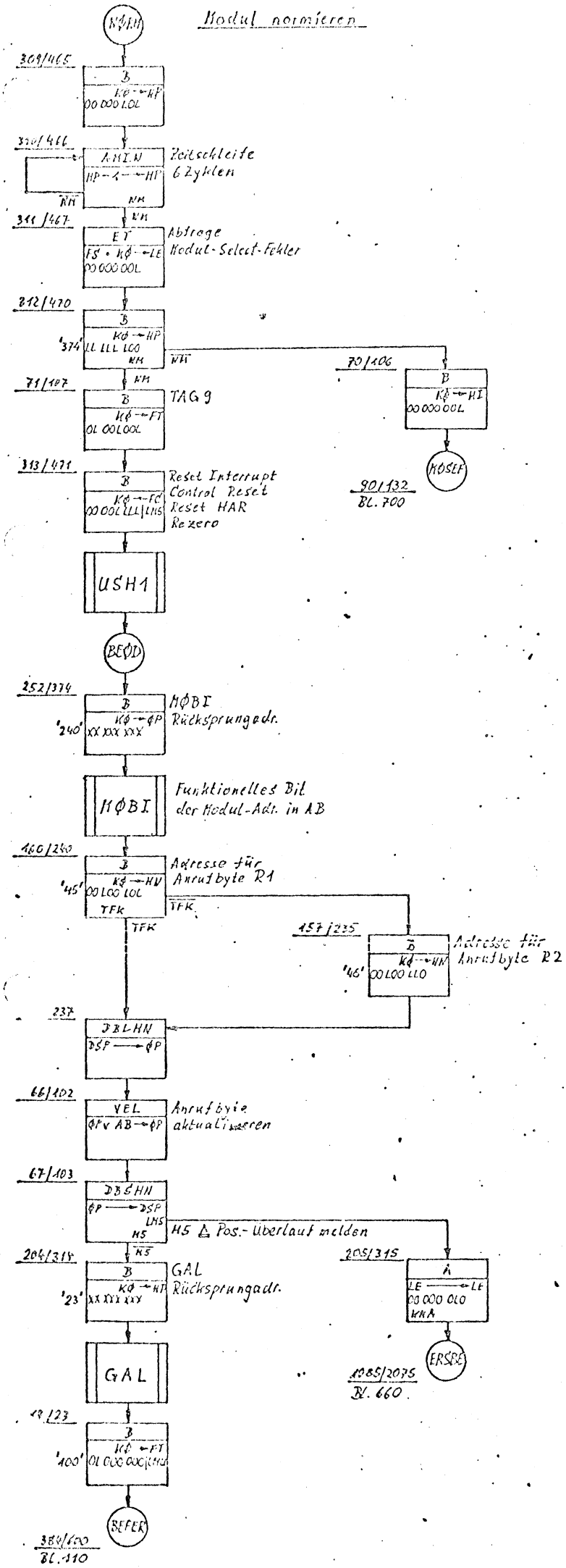
Diagnostik-Programm-Laden

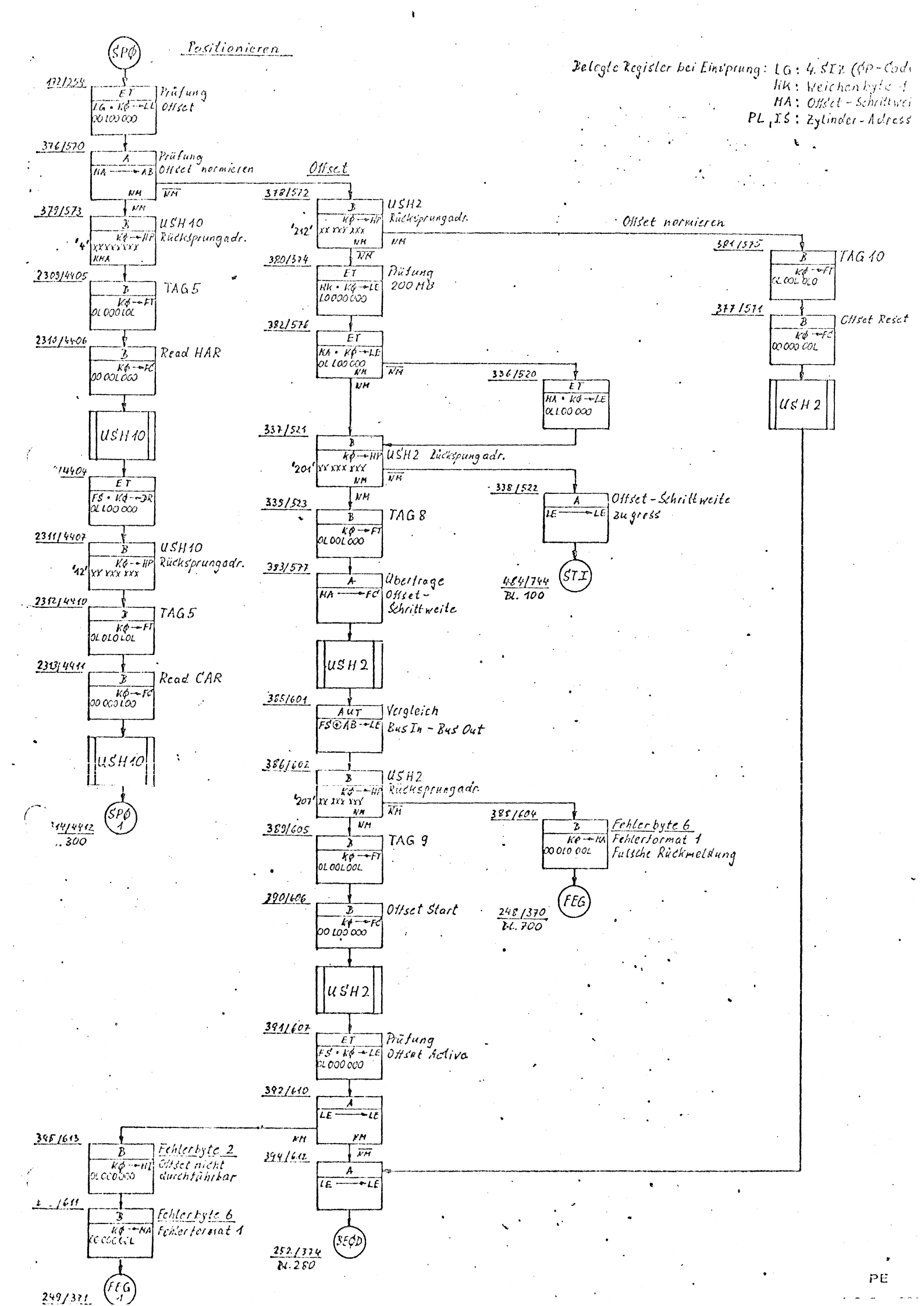


PE

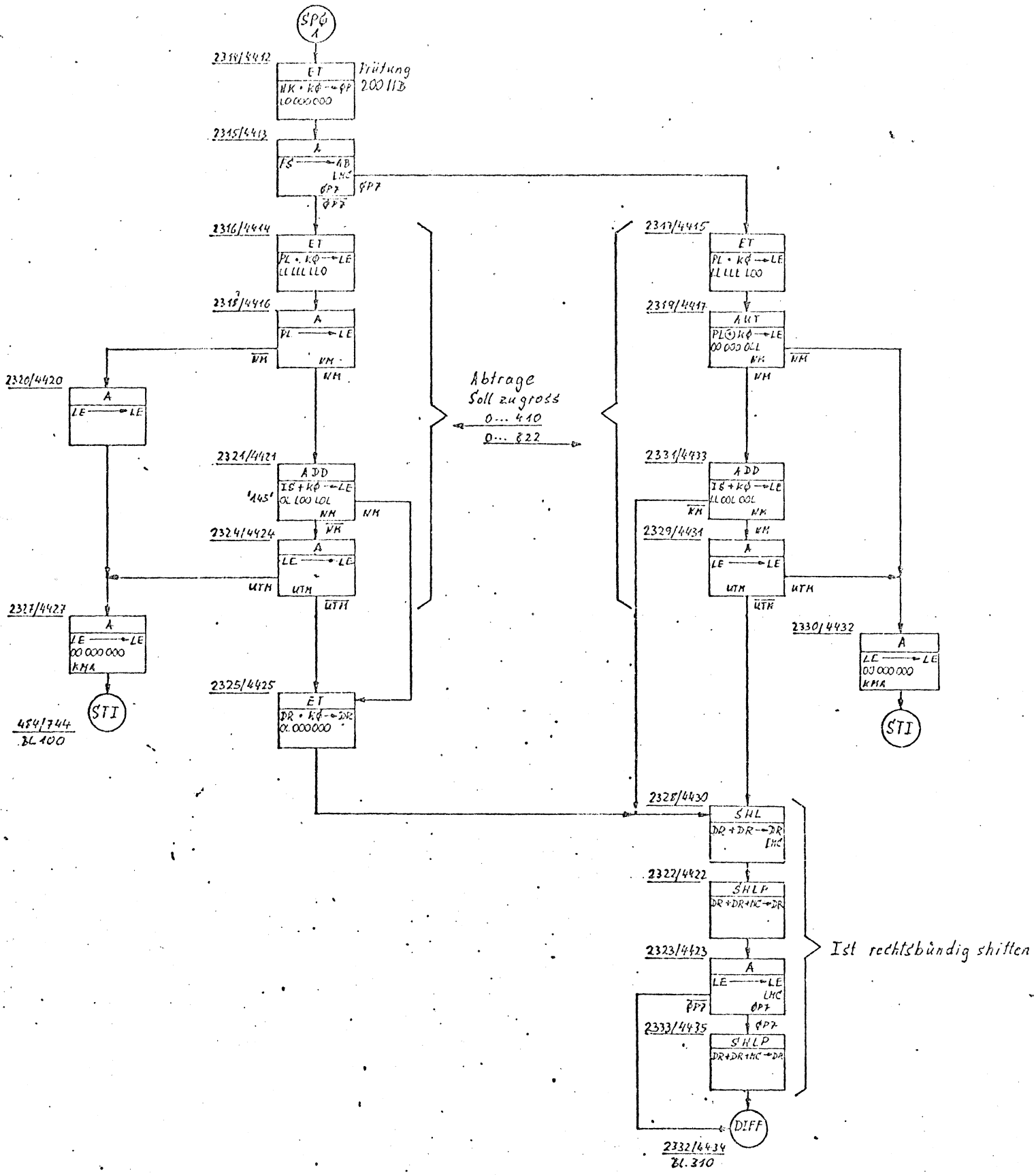

ZMFN
1191/2247
ET
AB + KØ → AB
00 000 011
ZBIT
1201/2267
B
KØ → FR
(9) 00 001 001
1244/2334
SHL
AB + AB → AB
MM
MM
1245/2335
AMIN
FR − 1 → FR
MTH
MTH
1246/2336
ADD
ØP + KØ → ØP
00 000 001
MM MM
MM
MM
RFAB A
1223/2302
ET
AB + AA → AE
00 111 100
Z+1
1247/2301
1247/2337
ADD
HE + KØ → HE
00 000 001
HØRE
Prüfung Parity bits
HØRE
1 DBLKØ DSP → AB 00 001 010 Spr. → ZBIT 1224/2310
2 DBLKØ DSP → AB 00 001 011 Spr. → ZBIT 1225/2311
3 DBLKØ DSP → AB 00 001 100 Spr. → BLEND 1226/2312
4 B KØ → ØP 00 000 000 Spr. → Z+1 / P2F (ØPO / ØPO) 1227/2313
5 DBLKØ DSP → AB 00 001 100 Spr. → BLEND 1 1228/2314
6 DBLKØ DSP → AB 00 001 101 Spr. → ZBIT 1229/2315
7 DBLKØ DSP → AB 00 001 110 Spr. → ZBIT 1230/2316
8 DBLKØ DSP → AB 00 001 111 Spr. → ZBIT 1231/2317
9 DBLKØ DSP → AB 00 010 000 Spr. → ZBIT 1232/2320
10 B * KØ → HP XX XXX XXX Spr. → SCHR / P1F (ØPO / ØPO) 1233/2321
* ZMFN Rücksprungadr.
SCHR
1220/2304
ADD
HD + 1 → HD
DPL
10. DPL
1234/2322
A
LE → LE
MC
Nach dem 10. DPL ist ein Leerbefehl nötig
MC
1237/2325
B
KØ → KR
11 100 000
STP, AF, EMP
DFLN
1189/2245
BL. 260
1236/2324
B
KØ → AB
10 010 000
Hardwarefehler
P1F
1222/2306
B
KØ → AB
10 000 000
Parityfehler
P2F
1219/2303
B
KØ → AB
10 000 000
Parityfehler
ALFE
1203/2263
1248/2340
DBSKØ
AB → DSP
00 000 110
Fehlerbyte 6 laden
1235/2323
B
KØ → AB
00 000 000
1238/2326
B
KØ → HN
00 000 110
1249/2330
AMIN
HN − 1 → HN
Fehlerbytes 0...5 laden
1255/2327
DBSHN
AB → DSP
MM
MM
1241/2331
B
KØ → HC
10 000 000
Fehlereingriffs-zeichen
DPF
1084/2074

PE

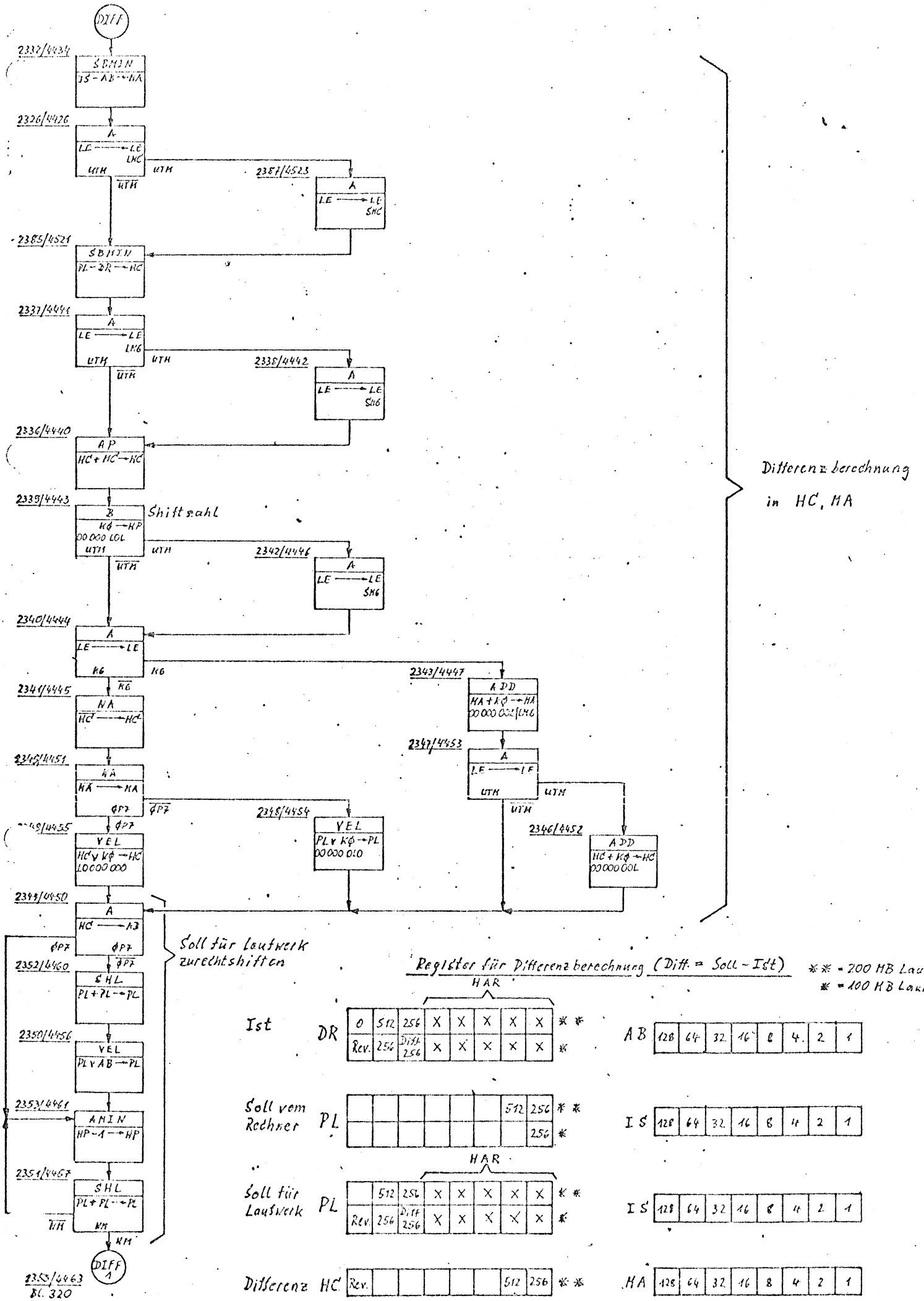

MODUL
Modul normieren
Belegte Register bei Einsprung: LG: 4.STZ (ΦD-Code)
HK: Weichenbyte 1
309/465
B
KΦ → HP
00 000 LOL
320/466
ANIN
Zeitschleife
6 Zyklen
HP - 1 → HP
NH
NH
341/467
ET
Abfrage
Modul-Select-Fehler
FS • KΦ → LE
00 000 00L
342/470
B
KΦ → HP
'374'
LL LLL LOO
NH
NH
70/106
B
KΦ → HI
00 000 00L
MOSEF
71/107
NH
B
TAG 9
KΦ → FT
OL OO LOOL
313/471
B
Reset Interrupt
Control Reset
Reset HAR
Rezero
KΦ → FC
00 00L LLL LHS
901/132
BL. 700
USH1
BEΦD
252/334
B
MΦBI
Rückspungadr.
KΦ → ΦP
'240'
XX XXX XXX
MΦBI
Funktionelles Bit
der Modul-Adr. in AB
460/240
B
Adresse für
Anrufbyte R1
KΦ → HV
'45'
00 LOO LOL
TFK
TFK
TFK
457/235
B
Adresse für
Anrufbyte R2
KΦ → HK
'46'
00 LOO LLO
237
DBLHN
DSP → ΦP
66/102
VEL
Anrufbyte
aktualisieren
ΦFV AB → ΦP
67/103
DBSHN
ΦP → DSP
LHS
HS
HS △ Pos.-Überlauf melden
HS
204/314
B
GAL
Rückspungadr.
KΦ → HT
'23'
XX XXX XXY
205/315
A
LE → LE
00 000 OLO
VRR
ERSBE
1085/2075
BL. 660
GAL
49/123
B
KΦ → FT
'100'
OL 000 00O LHS
BEFEH
384/600
BL. 110

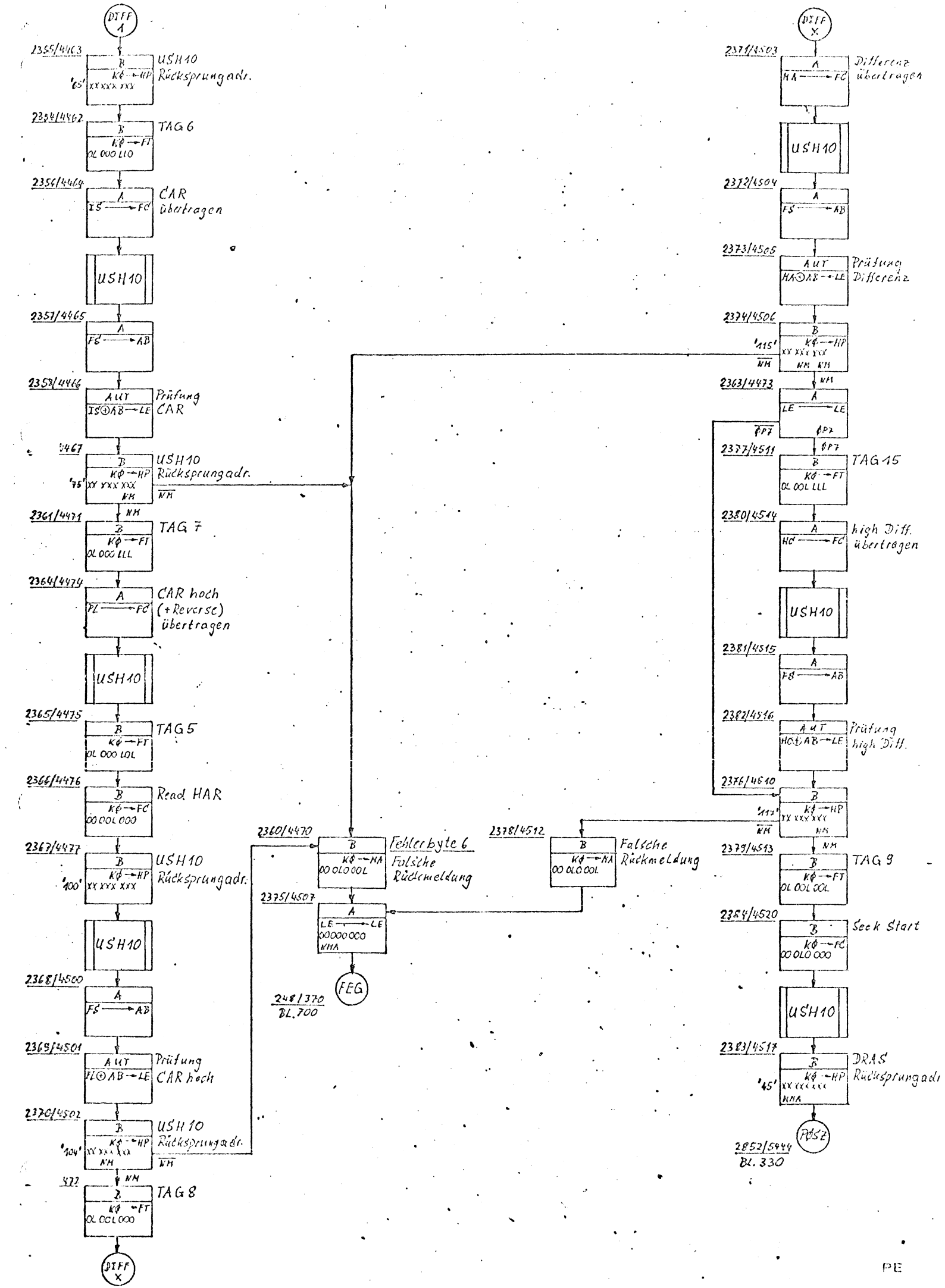

SPØ
Positionieren
Belegte Register bei Einsprung: LG : 4. SIZ (OP-Cod
HK : Weichenbyte 1
HA : Offset - Schrittwei
PL, IS : Zylinder - Adress
172/254
ET
LG • KØ → LL
00 100 000
Prüfung Offset
376/570
A
HA → AB
NM
Prüfung Offset normieren
NM
Offset
378/572
B
KØ → HP
XX XXX XXX
NM
USH2 Rücksprungadr.
'212'
Offset normieren
379/573
B
KØ → HP
XXXXXXXX
HHA
USH10 Rücksprungadr.
'4'
380/574
ET
HK • KØ → LE
10000000
Prüfung 200 MB
2309/4405
B
KØ → FT
01 000 101
TAG 5
382/576
ET
HA • KØ → LE
01 100 000
NM
336/520
ET
HA • KØ → LE
01 100 000
381/575
B
KØ → FT
01 001 010
TAG 10
377/571
B
KØ → FC
00 000 001
Offset Reset
2310/4406
B
KØ → FC
00 001 000
Read HAR
USH 2
USH10
337/521
B
KØ → HP
XX XXX XXX
NM
USH2 Rücksprungadr.
'201'
14404
ET
FS • KØ → DR
01 100 000
338/522
A
LE → LE
Offset - Schrittweite zugross
2311/4407
B
KØ → HP
XX XXX XXX
NM
USH10 Rücksprungadr.
'12'
339/523
B
KØ → FT
01 001 000
TAG 8
383/577
A
HA → FC
Übertrage Offset - Schrittweite
464/744
Bl. 100
ST.I
2312/4410
B
KØ → FT
01 010 101
TAG 5
USH 2
2313/4411
B
KØ → FC
00 000 100
Read CAR
385/601
AUT
FS ⊕ AB → LE
Vergleich Bus In - Bus Out
USH10
386/602
B
KØ → HP
XX XXX XXX
NM
USH2 Rücksprungadr.
'207'
SPØ 1
144/4412
..300
385/604
B
KØ → HA
00 010 001
Fehlerbyte 6 Fehlerformat 1 Falsche Rückmeldung
389/605
B
KØ → FT
01 001 001
TAG 9
FEG
248/370
Bl. 700
390/606
B
KØ → FC
00 100 000
Offset Start
USH 2
391/607
ET
FS • KØ → LE
01 000 000
Prüfung Offset Aktiv
392/610
A
LE → LE
NM
NM
395/613
B
KØ → HI
01 000 000
Fehlerbyte 2 Offset nicht durchführbar
394/612
A
LE → LE
./611
B
KØ → HA
FC 000 001
Fehlerbyte 6 Fehlerformat 1
FEG 1
249/371
SEQD
252/374
Bl. 280

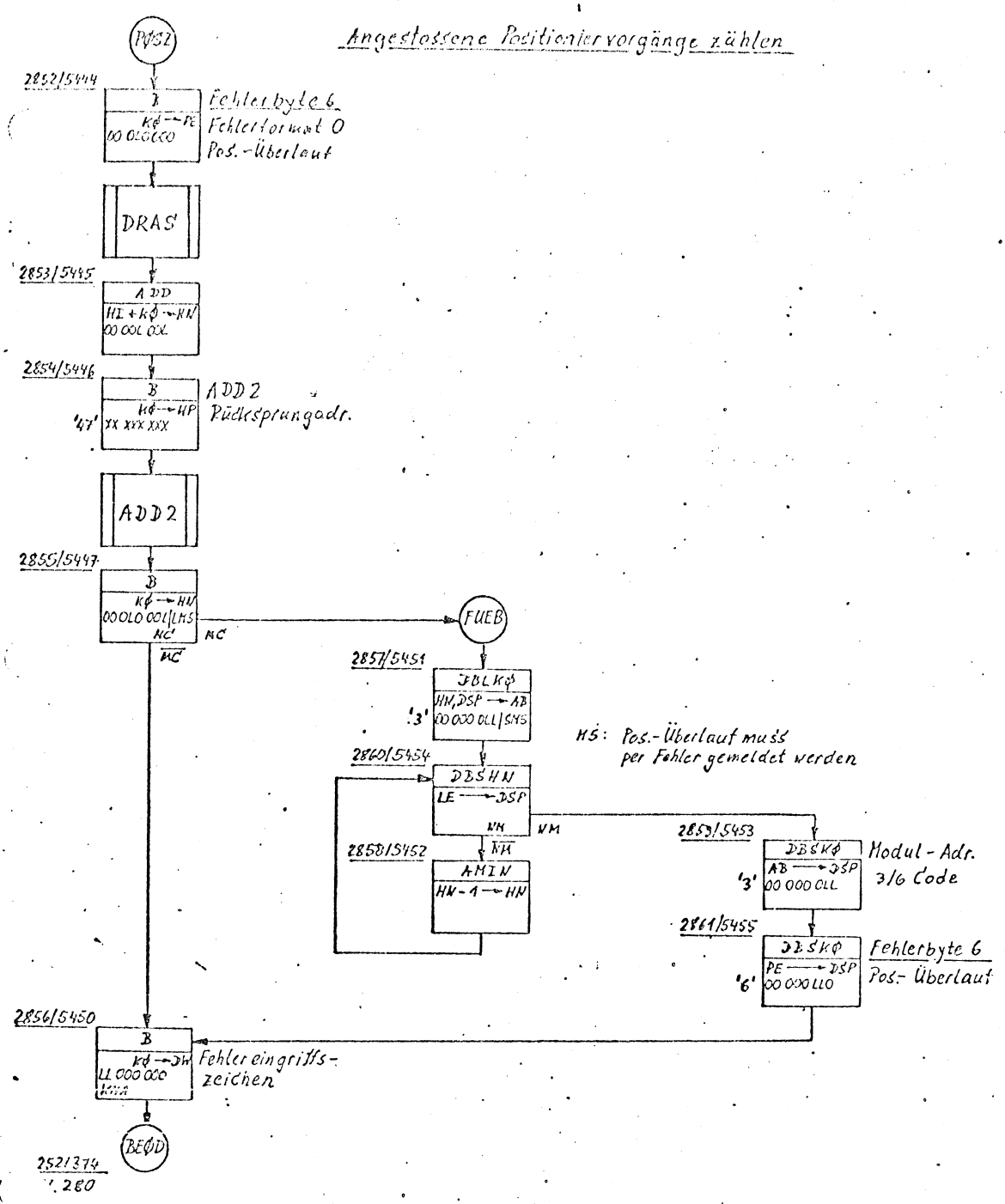
PE

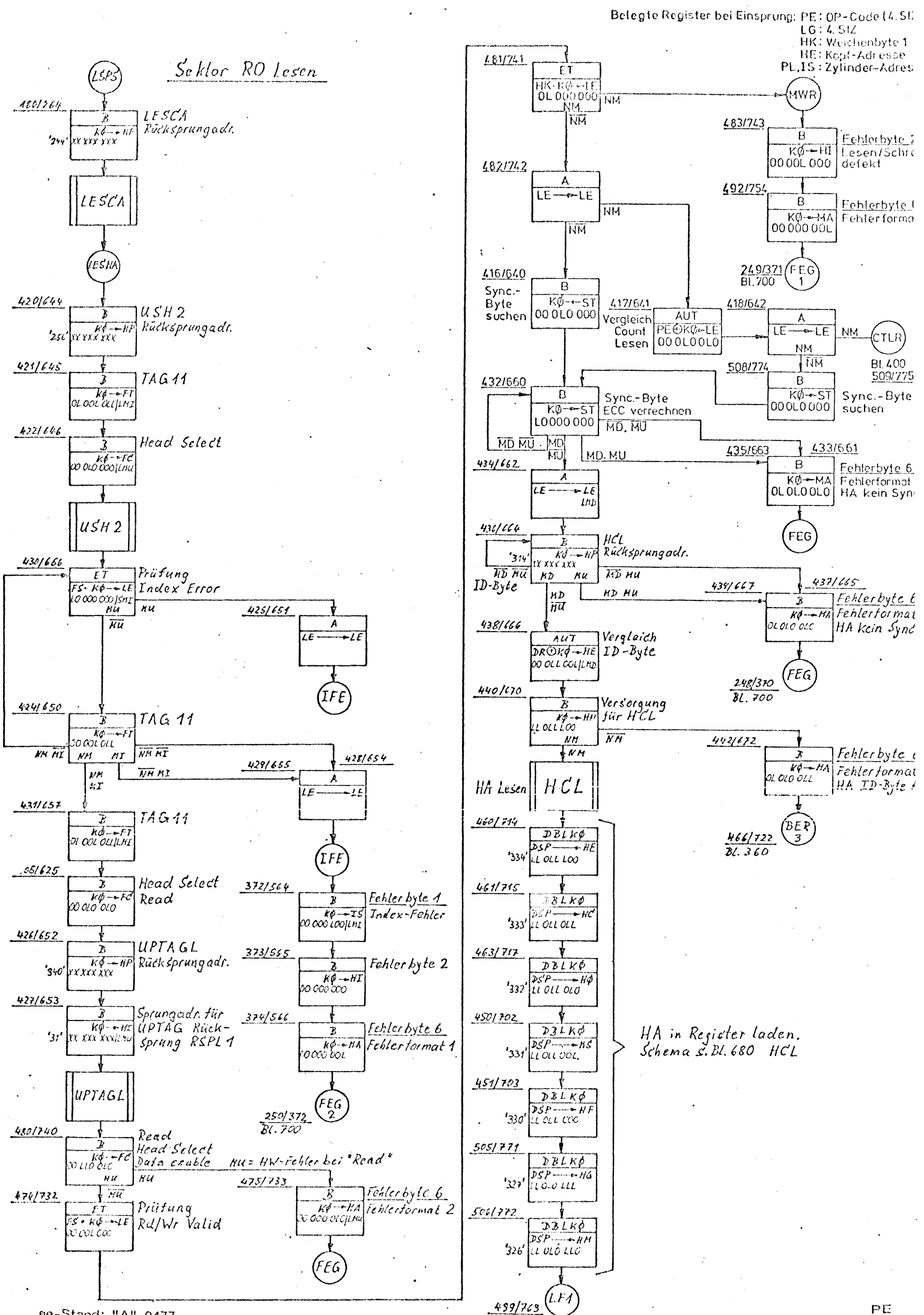

SPÜ A
2314/4412
ET
UK + KÜ → QP
L0 000 000
Prüfung 200 IIB
2315/4413
L
FS → AB
LHC
ØP7
ØP7
2316/4414
ET
PL + KÜ → LE
LL LLL LL0
2317/4415
ET
PL + KÜ → LE
LL LLL L00
2318/4416
A
PL → LE
UM
UM
2319/4417
AUT
PL ⊕ KÜ → LE
00 000 0LL
UM
UM
2320/4420
A
LE → LE
Abtrage
Soll zu gross
0 ... 410
0 ... 822
2321/4421
ADD
IS + KÜ → LE
'445'
0L L00 L0L
UM
UM
2331/4433
ADD
IS + KÜ → LE
LL L0L 0LL
UM
UM
2324/4424
A
LE → LE
UTM
UTM
2329/4431
A
LE → LE
UTM
UTM
2327/4427
A
LE → LE
00 000 000
KMA
2330/4432
A
LE → LE
00 000 000
KMA
STI
STI
484/744
BL 400
2325/4425
ET
DR + KÜ → DR
0L 000 000
2328/4430
SHL
DR + DR → DR
LHC
2322/4422
SHLP
DR + DR + HC → DR
2323/4423
A
LE → LE
LHC
ØP7
ØP7
2333/4435
SHLP
DR + DR + HC → DR
Ist rechtsbündig shiften
DIFF
2332/4434
BL 310

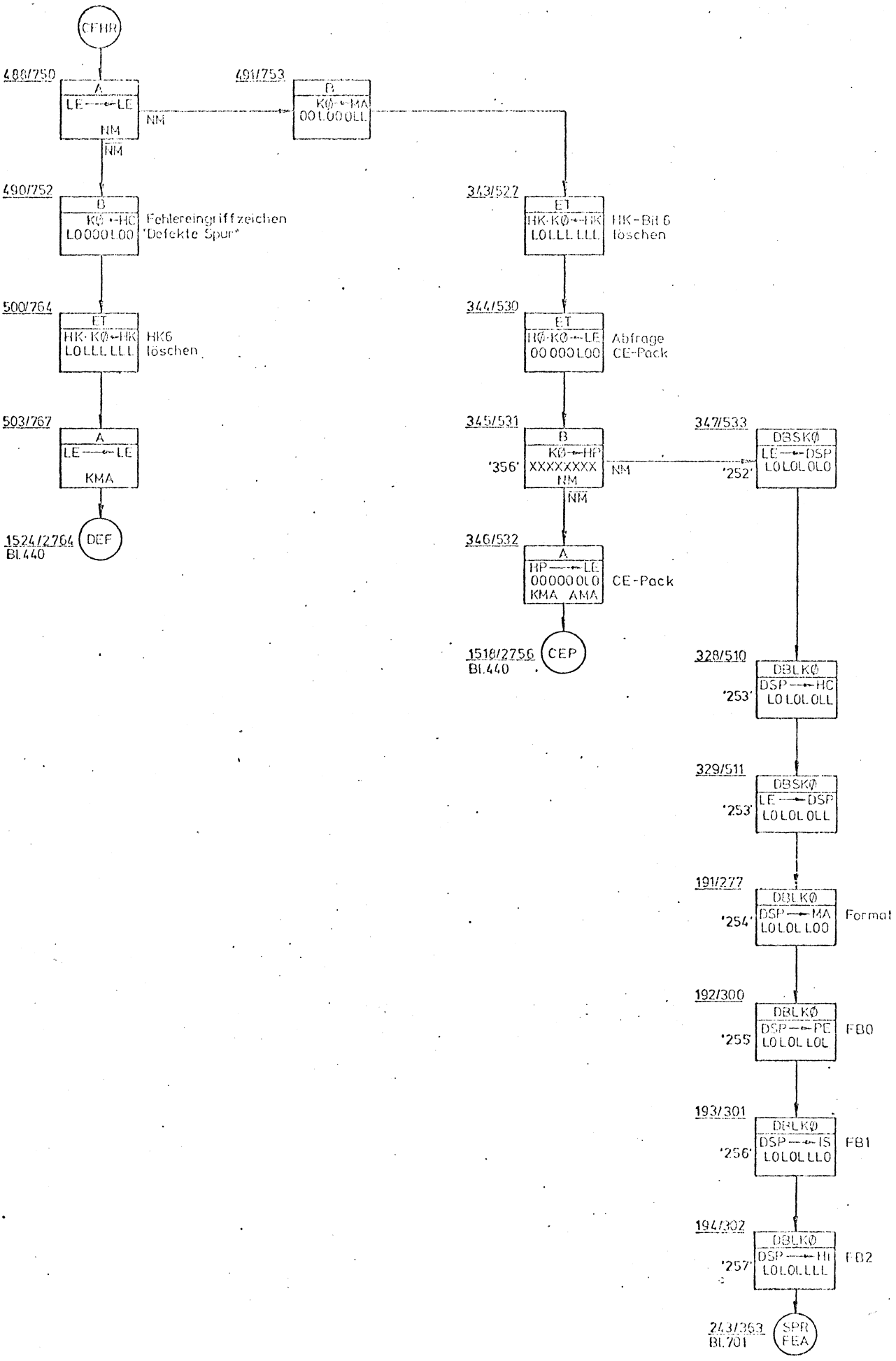
PE


DIFF
2332/4434
SBMIN
IS - AB → HA
2326/4426
A
LE → LE
UTH
2381/4523
SNC
2385/4521
SBMIN
PL - DR → HC
2337/4441
2338/4442
SN6
2336/4440
AP
HC + HC → HC
2339/4443
B
Shiftzahl
KΦ → HP
00 000 001
2342/4446
2340/4444
KG
2343/4447
ADD
HA + KΦ → HA
2341/4445
NA
HC → HC
2347/4453
2345/4451
HA → HA
ΦP7
2348/4454
VEL
PL ∨ KΦ → PL
00 000 010
2346/4452
ADD
HC + KΦ → HC
00 000 001
2344/4450
HC → A3
2352/4460
SHL
PL + PL → PL
2350/4456
VEL
PL ∨ AB → PL
2353/4461
AMIN
HP - 1 → HP
2354/4467
NM
2355/4463
Bl. 320
Differenzberechnung in HC, HA
Soll für Laufwerk zurechtshiften
Register für Differenzberechnung (Diff. = Soll - Ist)
** = 200 MB Laufwerk
* = 100 MB Laufwerk
HAR
Ist
DR
Soll vom Rechner
PL
Soll für Laufwerk
Differenz HC
AB
IS
HA
Rev.
Diff 256
entfällt *


DIFF A
2355/4463
B
KØ → HP
'65' XX XXX XXX
USH10 Rücksprungadr.
2354/4462
B
KØ → FT
OL 000 110
TAG 6
2356/4464
A
IS → FC
CAR übertragen
USH10
2357/4465
A
FS → AB
2358/4466
AUT
IS ⊕ AB → LE
Prüfung CAR
2467
B
KØ → HP
'75' XX YXX XXX
NH
USH10 Rücksprungadr.
NH
2361/4471
NH
B
KØ → FT
OL 000 111
TAG 7
2364/4474
A
PL → FC
CAR hoch (+Reverse) übertragen
USH10
2365/4475
B
KØ → FT
OL 000 101
TAG 5
2366/4476
B
KØ → FC
00 001 000
Read HAR
2367/4477
B
KØ → HP
'100' XX YYX XXX
USH10 Rücksprungadr.
USH10
2368/4500
A
FS → AB
2369/4501
AUT
PL ⊕ AB → LE
Prüfung CAR hoch
2370/4502
B
KØ → HP
'104' XX XXX XXX
NH
USH10 Rücksprungadr.
NH
522
NH
B
KØ → FT
OL 001 000
TAG 8
DIFF X
2360/4470
B
KØ → HA
00 010 00L
Fehlerbyte 6 Falsche Rückmeldung
2375/4507
A
LE → LE
00000 000
HHA
FEG
248/370
BL. 700
2378/4512
B
KØ → HA
00 010 00L
Falsche Rückmeldung
DIFF X
2371/4503
A
HA → FC
Differenz übertragen
USH10
2372/4504
A
FS → AB
2373/4505
AUT
HA ⊕ AB → LE
Prüfung Differenz
2374/4506
B
KØ → HP
'115' XX XXX XXX
NH NH
NH
2363/4473
NH
A
LE → LE
BP7
BP7
2377/4511
BP7
B
KØ → FT
OL 001 LLL
TAG 15
2380/4514
A
HC → FC
high Diff. übertragen
USH10
2381/4515
A
FB → AB
2382/4516
AUT
HC ⊕ AB → LE
Prüfung high Diff.
2376/4610
B
KØ → HP
'112' XX XXX XXX
NH
NH
2379/4513
NH
B
KØ → FT
OL 001 001
TAG 9
2384/4520
B
KØ → FC
00 010 000
Seek Start
USH10
2383/4517
B
KØ → HP
'45' XX XXX XXX
HHA
Rücksprungadr.
PUSZ
2852/5444
BL. 330
PE

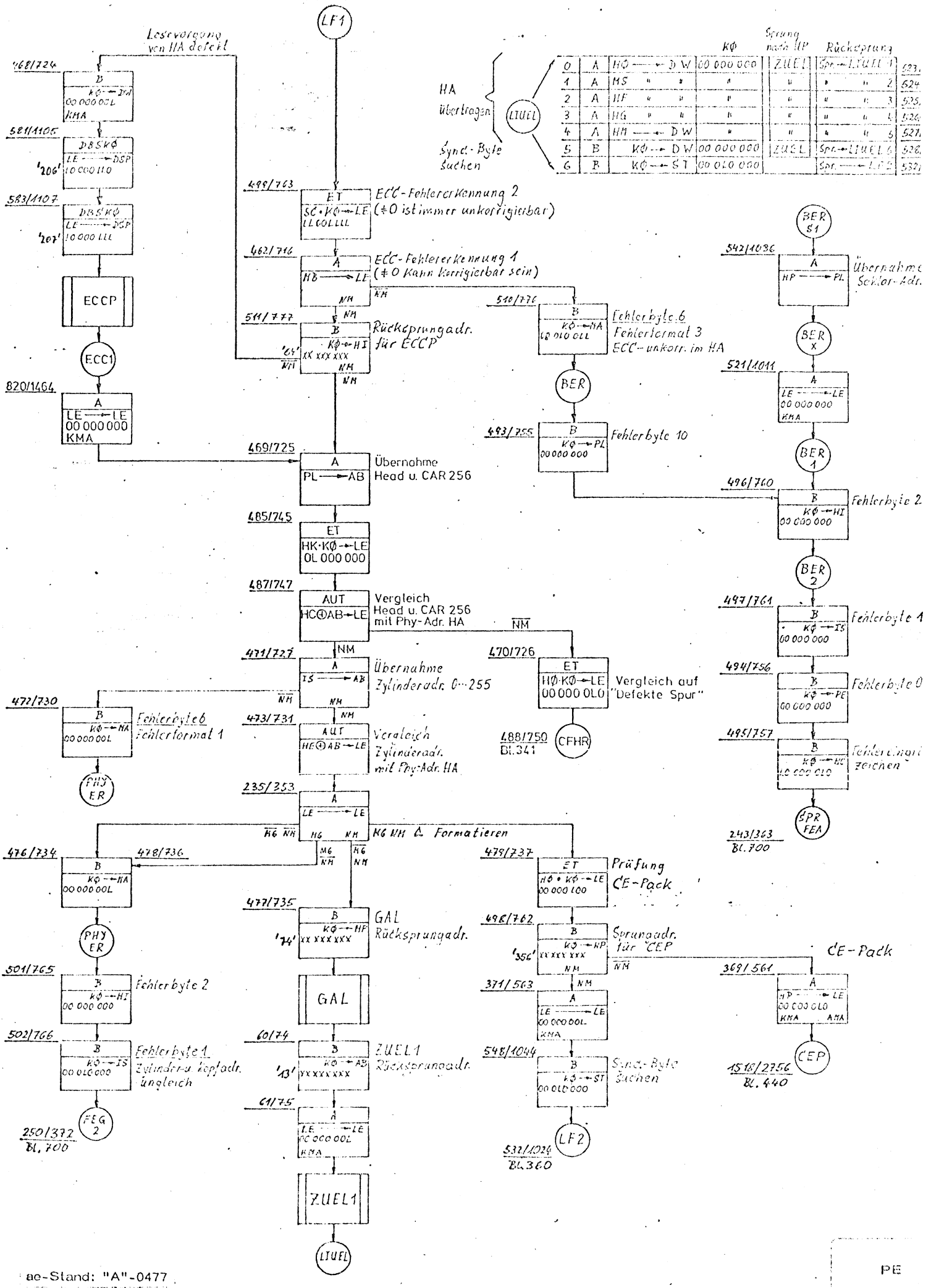

PØSZ
Angestossene Positionier vorgänge zählen
2852/5444
B
KØ → PE
00 010000
Fehlerbyte 6
Fehlerformat 0
Pos.-Überlauf
DRAS
2853/5445
ADD
HI + KØ → HN
00 001 00L
2854/5446
B
KØ → HP
'47' XX XXX XXX
ADD 2
Rücksprungadr.
ADD2
2855/5447
B
KØ → HN
000L0 00L/LHS
NC'
NC
NC
FUEB
2857/5451
JBLKØ
HM, DSP → AB
'3' 00 000 0LL/SHS
HS: Pos.-Überlauf muss per Fehler gemeldet werden
2860/5454
DBSHN
LE → DSP
NM
NM
2858/5452
NM
AMIN
HM-1 → HM
2859/5453
DBSKØ
AB → DSP
'3' 00 000 0LL
Modul-Adr. 3/6 Code
2861/5455
DBSKØ
PE → DSP
'6' 00 000 LL0
Fehlerbyte 6
Pos.-Überlauf
2856/5450
B
KØ → JW
LL 000 000
Fehlereingriffs-zeichen
BEØD
2521/374
'.280


Belegte Register bei Einsprung: PE: OP-Code (4. SI
LG: 4. SIZ
HK: Weichenbyte 1
HE: Kopf-Adresse
PL,IS: Zylinder-Adres
Sektor RO Lesen
LSPS
180/264
LESCA
Rücksprungadr.
LESCA
VESHA
420/644
USH 2
Rücksprungadr.
421/645
TAG 11
422/646
Head Select
USH 2
430/666
Prüfung
Index Error
425/651
IFE
424/650
TAG 11
428/654
429/655
IFE
431/657
TAG 11
Head Select
Read
426/652
UPTAGL
Rücksprungadr.
427/653
Sprungadr. für
UPTAG Rück-
sprung RSPL 1
UPTAGL
480/740
Read
Head Select
Data enable
474/732
Prüfung
Rd/Wr Valid
372/564
Fehlerbyte 1
Index-Fehler
373/565
Fehlerbyte 2
374/566
Fehlerbyte 6
Fehlerformat 1
250/372
Bl. 700
FEG 2
NM = HW-Fehler bei "Read"
475/733
Fehlerbyte 6
Fehlerformat 2
FEG
481/741
ET
MWR
482/742
483/743
Fehlerbyte
Lesen/Schr
defekt
492/754
Fehlerbyte
Fehlerforma
249/371
Bl. 700
FEG 1
416/640
Sync.-
Byte
suchen
417/641
Vergleich
Count
Lesen
418/642
CTLR
Bl. 400
509/775
508/774
Sync.-Byte
suchen
432/660
Sync.-Byte
ECC verrechnen
MD. MU
435/663
433/661
Fehlerbyte 6
Fehlerformat
HA kein Syn
434/662
436/664
HCL
Rücksprungadr.
ID-Byte
434/667
437/665
Fehlerbyte 6
Fehlerformat
HA kein Sync
438/666
Vergleich
ID-Byte
248/370
Bl. 700
FEG
440/670
Vers'orgung
für HCL
442/672
Fehlerbyte
Fehlerformat
HA ID-Byte
BER 3
466/722
Bl. 360
HA Lesen
HCL
460/714
461/715
463/717
450/702
451/703
505/771
506/772
HA in Register laden.
Schema s. Bl. 680 HCL
LF1
459/763
Bl. 350
ae-Stand: "A"-0477
PE
4.3.2 - 34

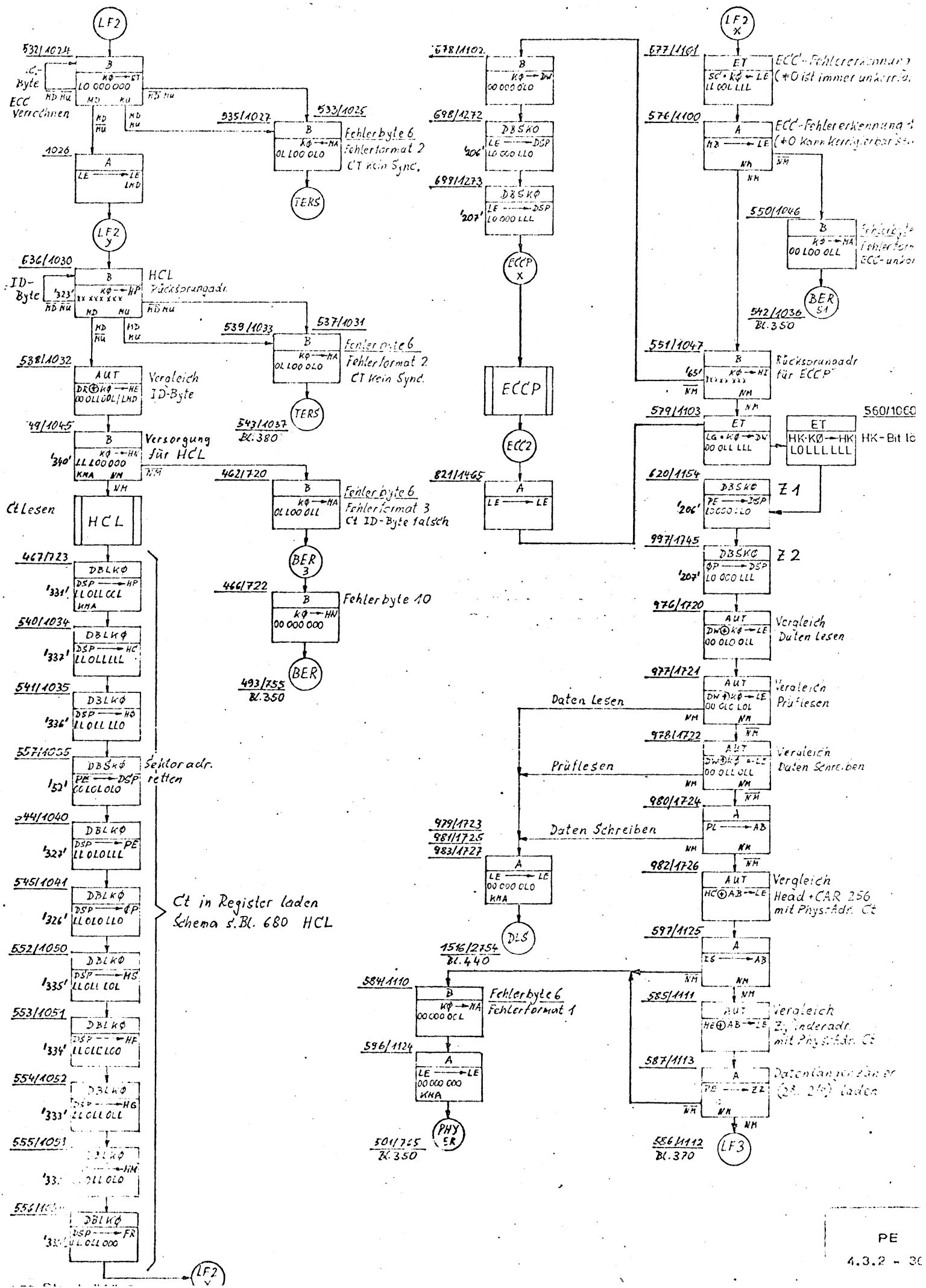


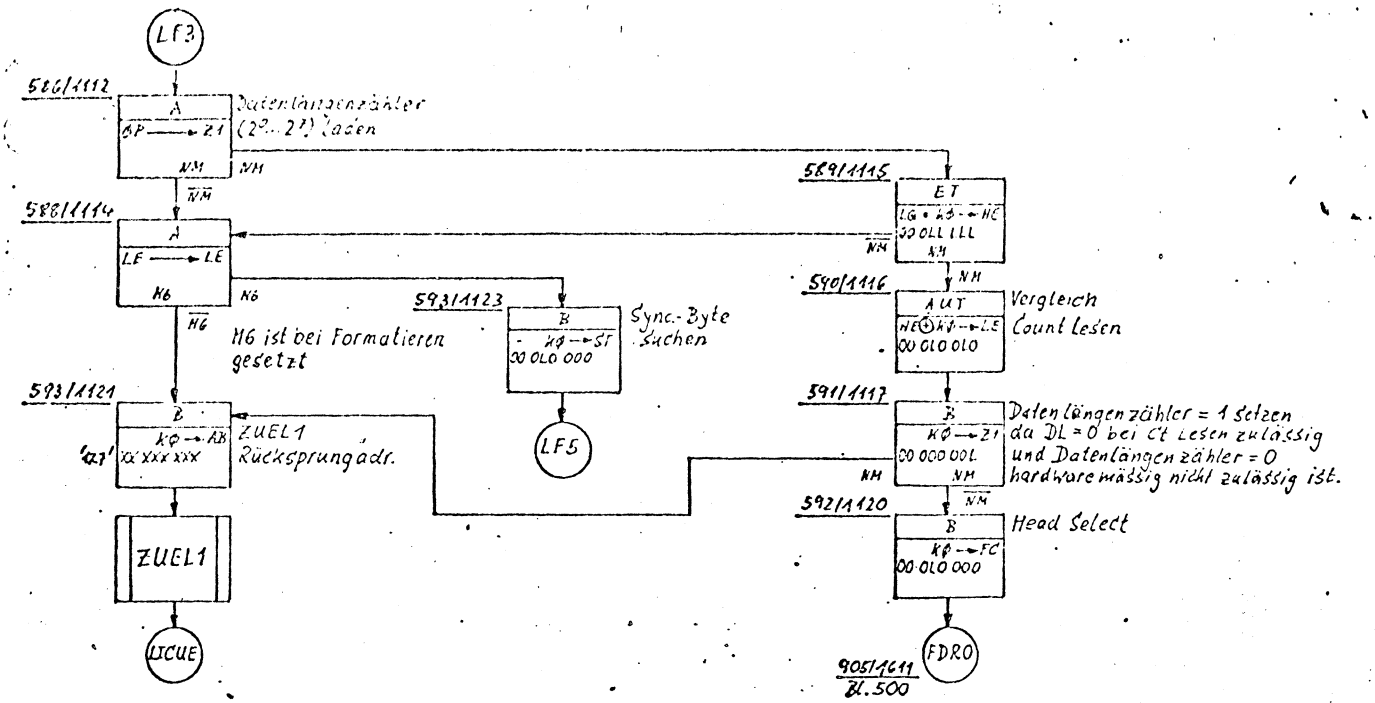
ae-Stand: "A"-0477

PE



ae-Stand: "A"-0477


LF2
532/1034
B
C.-Byte
ECC Verrechnen
1026
A
LF2 y
536/1030
ID-Byte
HCL Rücksprungadr.
538/1032
Vergleich ID-Byte
49/1045
Versorgung für HCL
CT lesen
HCL
467/723
540/1034
541/1035
557/1055
Sektoradr. retten
544/1040
545/1041
CT in Register laden
Schema s. Bl. 680 HCL
552/1050
553/1051
554/1052
555/1053
LF2 y
533/1025
535/1027
Fehlerbyte 6
Fehlerformat 2
CT kein Sync.
TERS
537/1031
539/1033
543/1037
Bl. 380
462/720
Fehlerbyte 6
Fehlerformat 3
CT ID-Byte falsch
BER 3
466/722
Fehlerbyte 10
BER
493/755
Bl. 350
678/1102
698/1272
699/1273
ECCP X
ECCP
ECC2
821/1465
LF2 x
677/1101
ECC-Fehlererkennung
576/1100
ECC-Fehlererkennung 1
550/1046
542/1036
Bl. 350
551/1047
Rücksprungadr für ECCP
579/1103
560/1060
HK-Bit lo
620/1154
Z 1
997/1745
Z 2
976/1720
Vergleich Daten lesen
977/1721
Vergleich Prüflesen
978/1722
Vergleich Daten Schreiben
980/1724
979/1723
981/1725
983/1727
Daten lesen
Prüflesen
Daten Schreiben
982/1726
Vergleich Head+CAR 256 mit Phys.-Adr. CT
597/1125
1516/2754
Bl. 440
584/1110
Fehlerbyte 6
Fehlerformat 1
596/1124
585/1111
Vergleich Zylinderadr. mit Phys.-Adr. CT
587/1113
Datenlängenzähler (28, 24) laden
501/765
Bl. 350
556/1112
Bl. 370
LF3

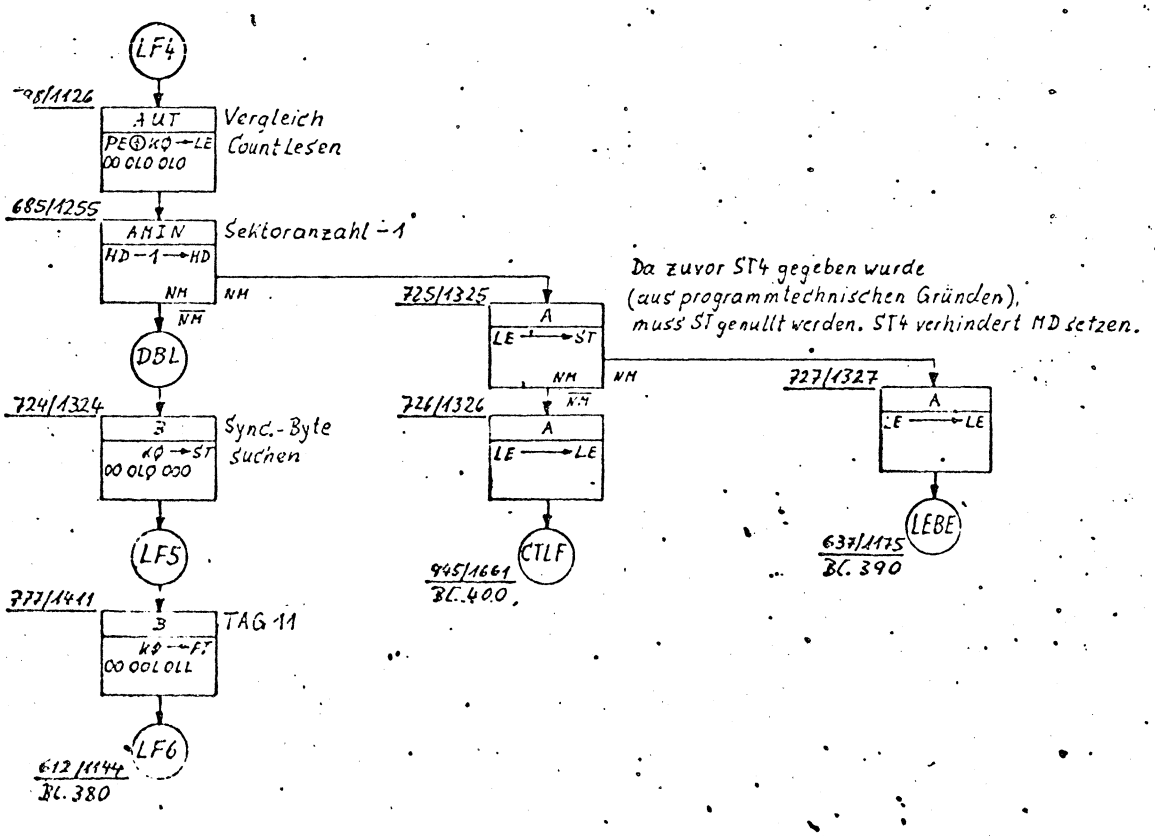


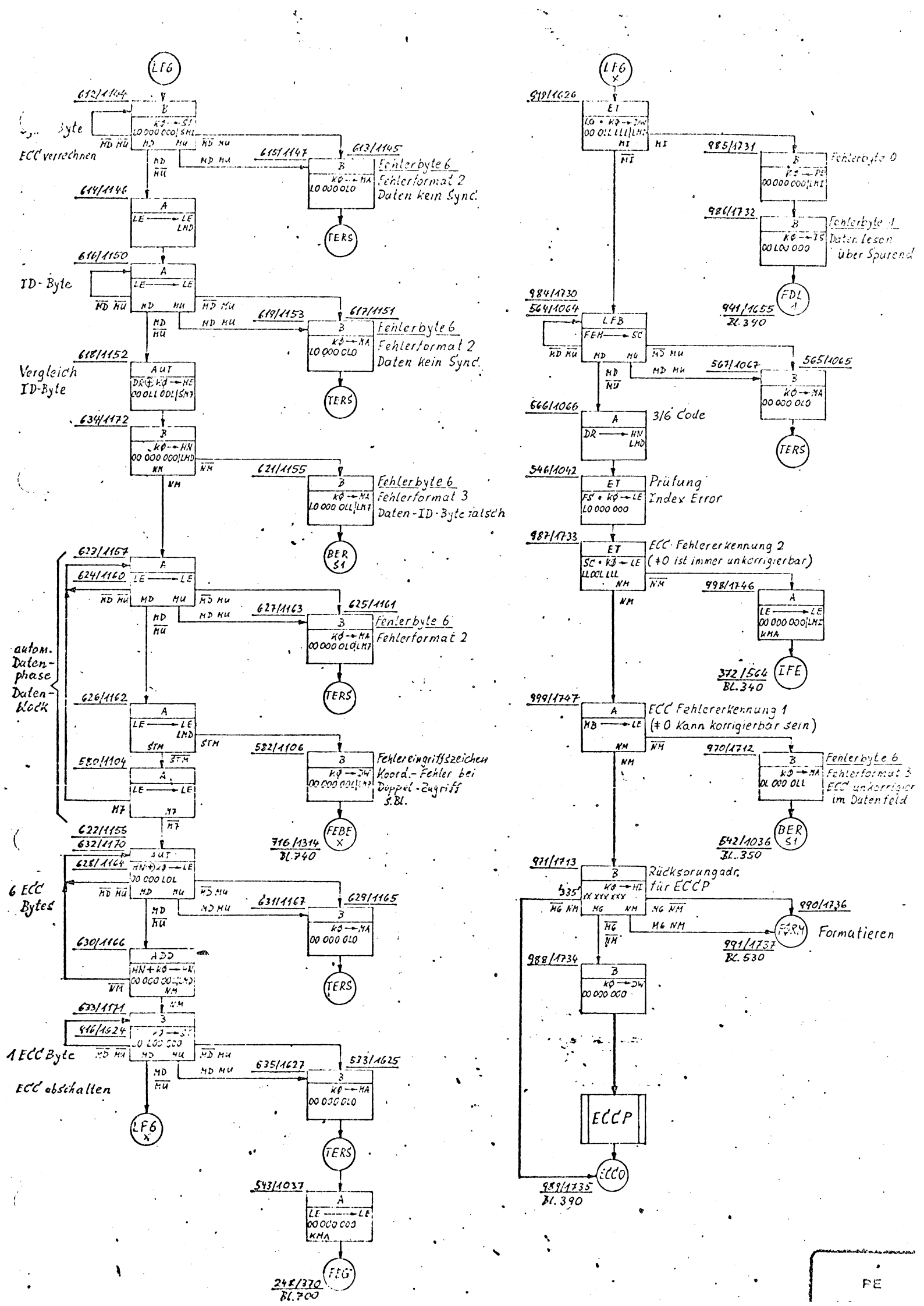
KØ				Sprung nach UP	Rücksprung nach	
0	A	HØ → DW	00 000 000	ZUEL	LICUE 1	599/1127
1	"	MS " "	"	"	" 2	600/1130
2	"	HF " "	"	"	" 3	601/1131
3	"	HG " "	"	"	" 4	602/1132
4	"	HM " "	"	"	" 5	603/1133
5	"	HP " "	"	"	" 6	604/1134
6	"	FR " "	"	"	" 7	605/1135
7	"	PE " "	"	"	" 8	606/1136
8	A	ØP → DW	"	"	" 9	607/1137
9	B	KØ → DW	00 000 000	"	" 10	608/1140
10	B	HØ " ST	00 010 000	"	" 11	609/1141
11	B	KØ " DW	00 000 000	ZUEL	LICUE 12	610/1142
12	ET	LG • KØ → PE	00 011 111		LF4	598/1126

LICUE

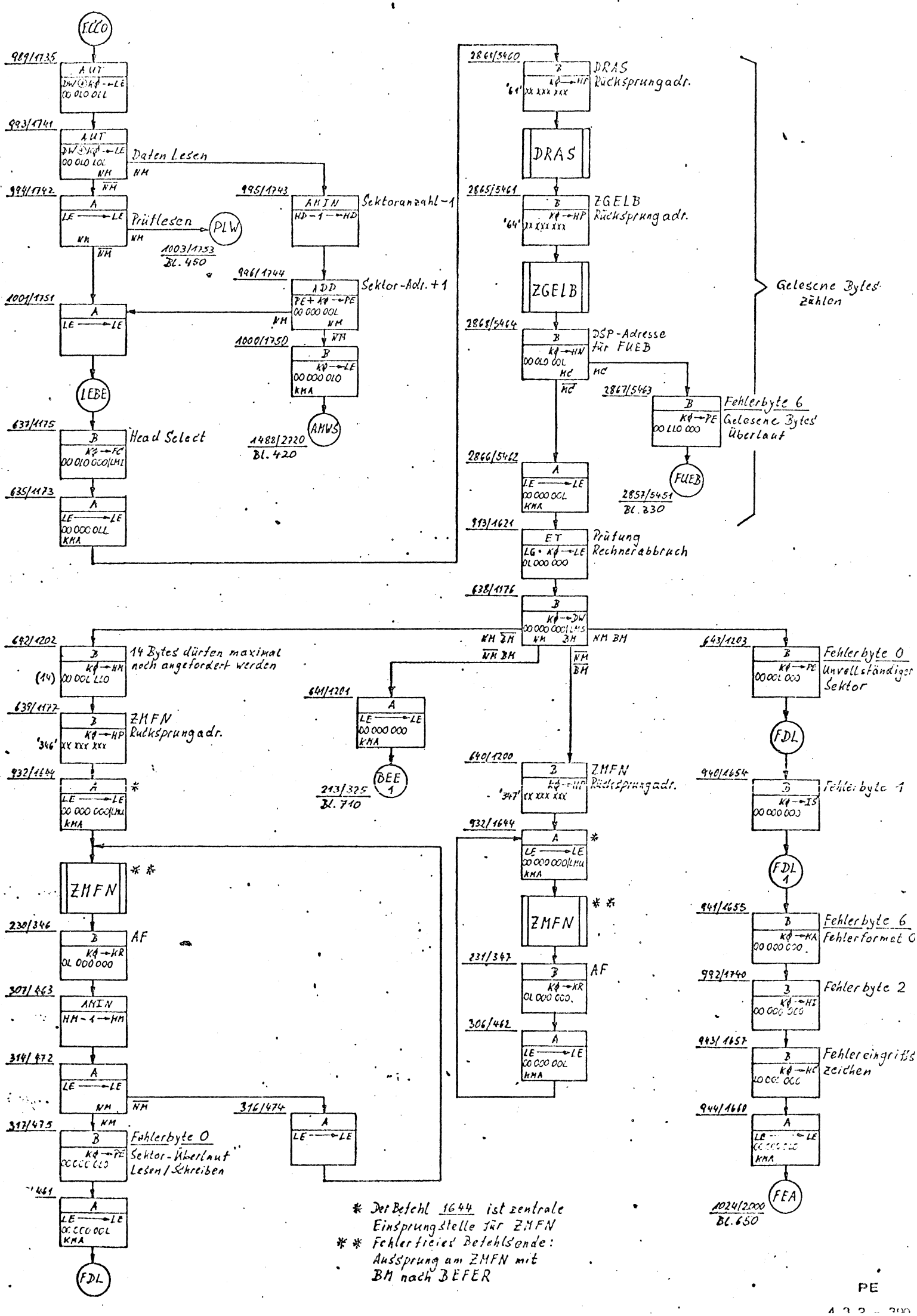
Ct übertragen

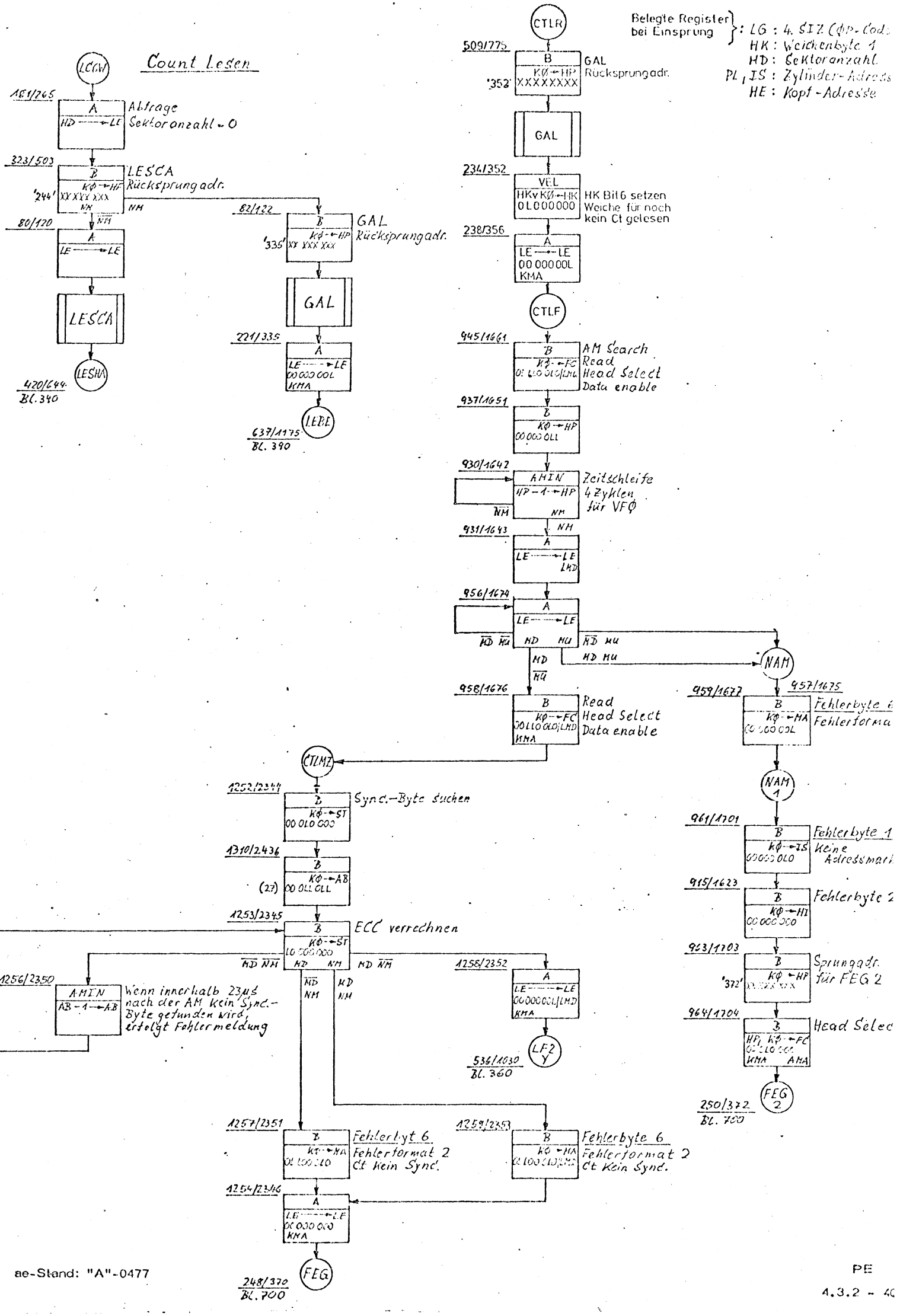
Synd.-Byte suchen




LF6
612/1144
B
Byte
ECC verrechnen
615/1147
613/1145
Fehlerbyte 6
Fehlerformat 2
Daten kein Sync
TERS
614/1146
A
616/1150
ID-Byte
619/1153
617/1151
Fehlerbyte 6
Fehlerformat 2
Daten kein Sync
TERS
618/1152
AUT
Vergleich
ID-Byte
634/1172
B
621/1155
Fehlerbyte 6
Fehlerformat 3
Daten-ID-Byte falsch
BER S1
633/1167
624/1160
627/1163
625/1161
Fehlerbyte 6
Fehlerformat 2
TERS
autom. Datenphase Datenblock
626/1162
582/1106
Fehlereingriffszeichen
Koord.-Fehler bei
Doppel-Zugriff
S.Bl.
580/1104
716/1314
Bl. 740
FEBE
622/1156
632/1170
628/1164
6 ECC Bytes
631/1167
629/1165
630/1166
633/1171
816/1624
1 ECC Byte
ECC abschalten
635/1627
633/1625
TERS
543/1037
248/370
Bl. 700
FEG
LF6
919/1626
EI
985/1731
Fehlerbyte 0
986/1732
Fehlerbyte 1
Daten lesen
über Spurende
FDL 1
941/1655
Bl. 340
984/1730
564/1064
LFB
567/1067
565/1065
666/1066
3/6 Code
546/1042
Prüfung
Index Error
987/1733
ECC-Fehlererkennung 2
(≠0 ist immer unkorrigierbar)
998/1746
IFE
372/564
Bl. 340
999/1747
ECC Fehlererkennung 1
(≠0 kann korrigierbar sein)
970/1712
Fehlerbyte 6
Fehlerformat 3
ECC unkorrigier
im Datenfeld
BER S1
642/1036
Bl. 350
971/1713
Rücksprungadr.
für ECCP
535
990/1736
FORM
Formatieren
991/1737
Bl. 530
988/1734
ECCP
ECCO
989/1735
Bl. 390

PE

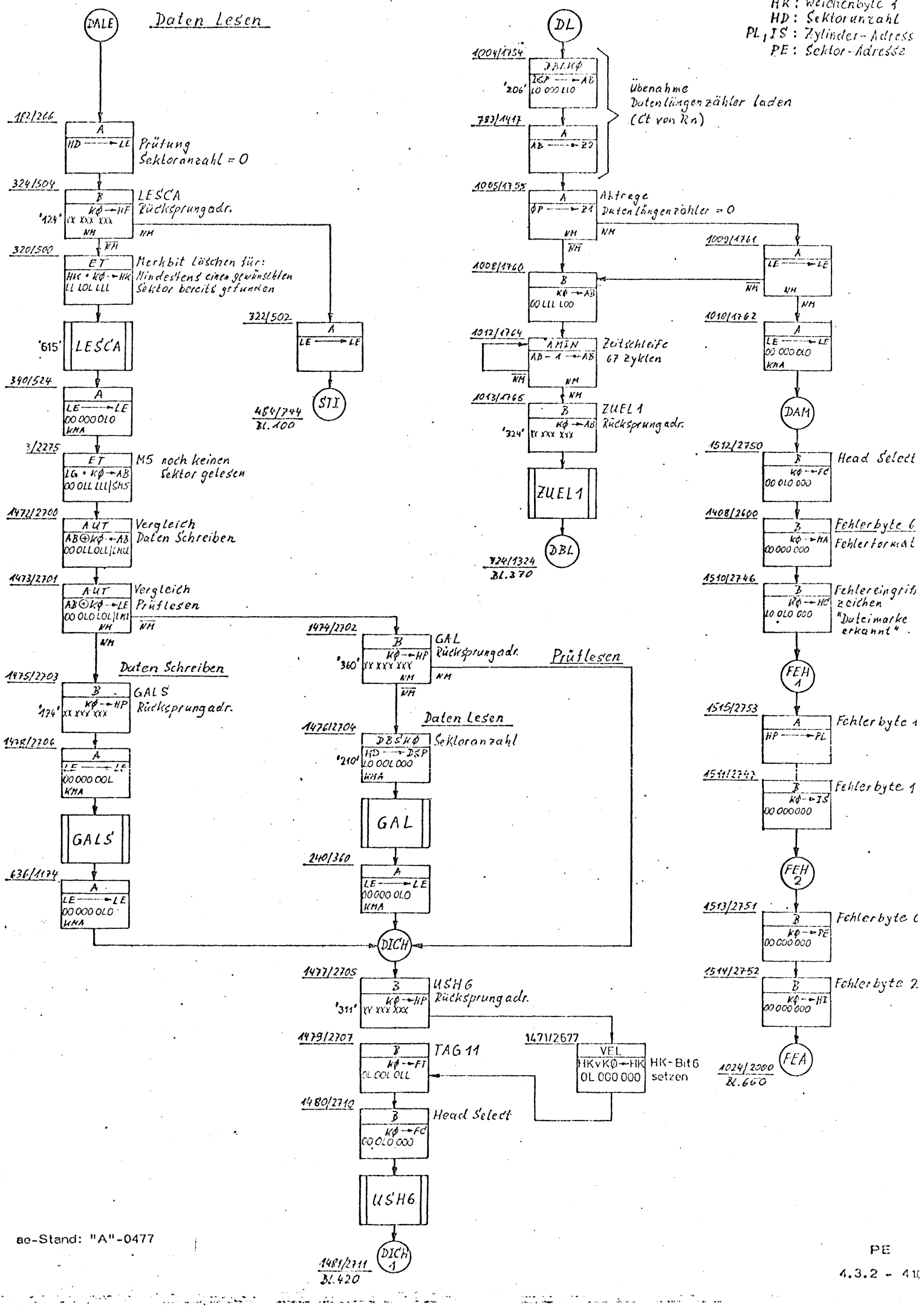

ECCO
989/1735
AUT
993/1741
AUT
Daten Lesen
994/1742
A
Prüflesen
PLW
1003/1753
Bl. 450
995/1743
AMIN
Sektoranzahl-1
996/1744
ADD
Sektor-Adr. +1
1001/1751
A
1000/1750
B
LEBE
637/1175
B
Head Select
635/1173
A
AMWS
1488/2720
Bl. 420
2864/5460
DRAS
Rücksprungadr.
DRAS
2865/5461
ZGELB
Rücksprungadr.
ZGELB
2868/5464
B
DSP-Adresse für FUEB
2867/5463
Fehlerbyte 6
Gelesene Bytes' Überlauf
FUEB
2857/5451
Bl. 830
Gelesene Bytes zählen
2866/5462
A
913/1621
ET
Prüfung Rechnerabbruch
638/1176
B
642/1202
14 Bytes dürfen maximal noch angefordert werden
(14)
639/1177
ZMFN
Rücksprungadr.
'346'
932/1644
A
ZMFN
230/346
AF
307/463
AMIN
314/472
A
317/475
Fehlerbyte 0
Sektor-Überlauf Lesen/Schreiben
461
FDL
641/1201
A
BEE 1
213/325
Bl. 710
316/474
A
640/1200
ZMFN
Rücksprungadr.
'347'
932/1644
A
ZMFN
231/347
AF
306/462
A
643/1203
Fehlerbyte 0
Unvollständiger Sektor
FDL
940/1654
Fehlerbyte 1
FDL 1
941/1655
Fehlerbyte 6
Fehlerformat 0
992/1740
Fehlerbyte 2
943/1657
Fehlereingriffszeichen
944/1660
A
FEA
1024/2000
Bl. 650
* Der Befehl 1644 ist zentrale Einsprungstelle für ZMFN
** Fehlerfreies Befehlsende: Ausssprung am ZMFN mit BM nach BEFER



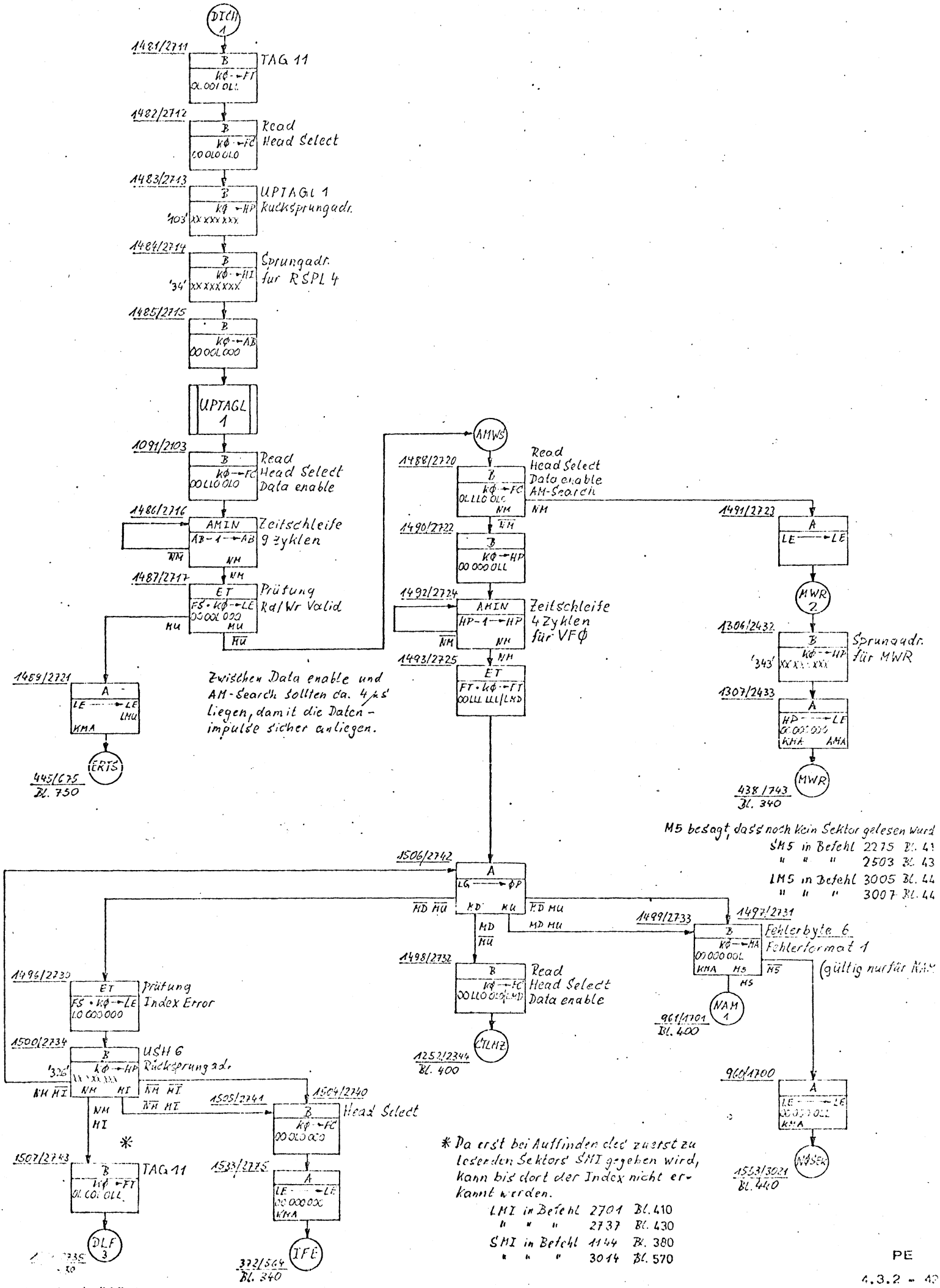
ae-Stand: "A"-0477

PE

Belegte Register bei Einsprung: LG : 4. SIZ (CP-Code
HK : Weichenbyte 1
HD : Sektoranzahl
PL, IS : Zylinder-Adress
PE : Sektor-Adresse



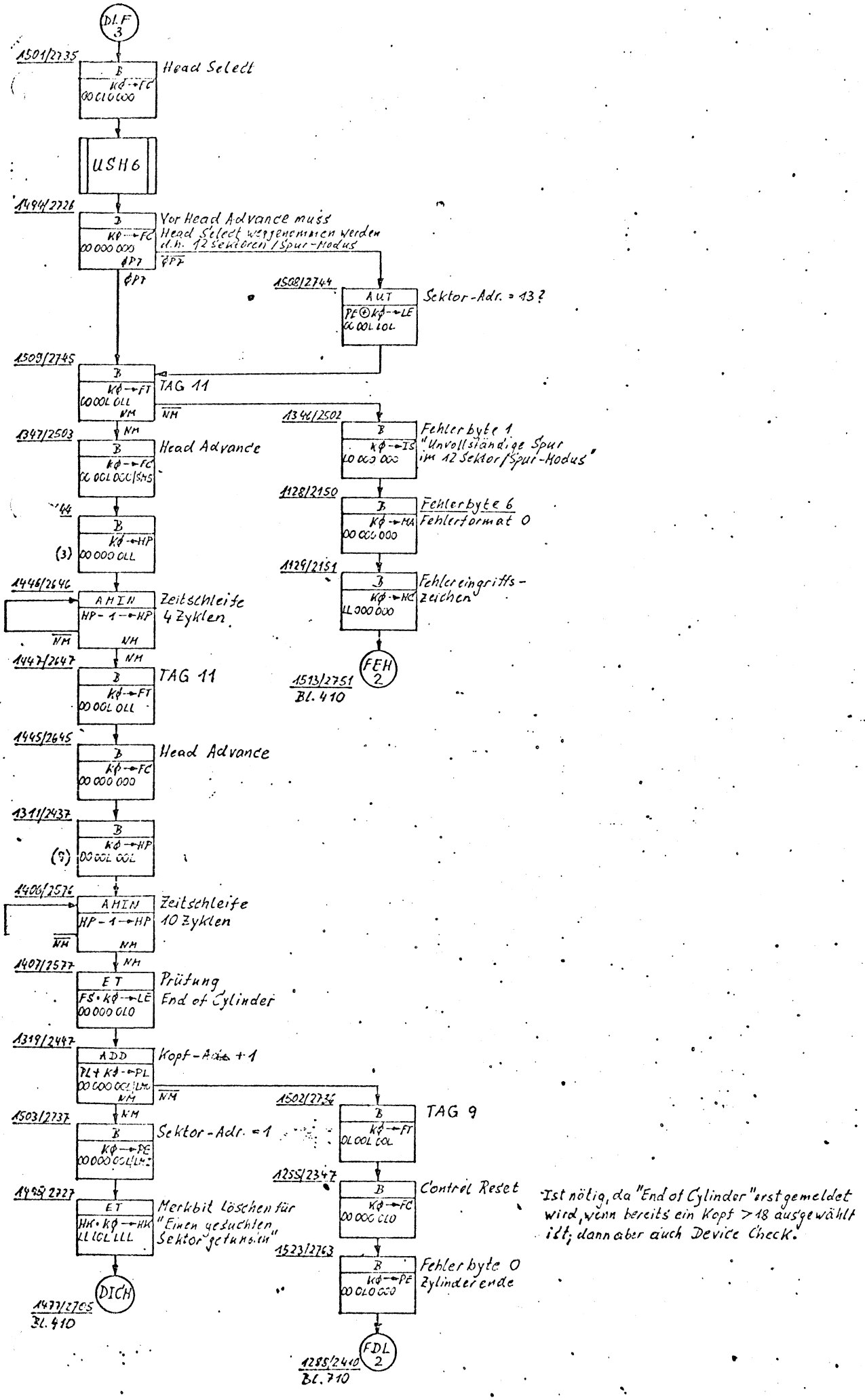
ae-Stand: "A"-0477

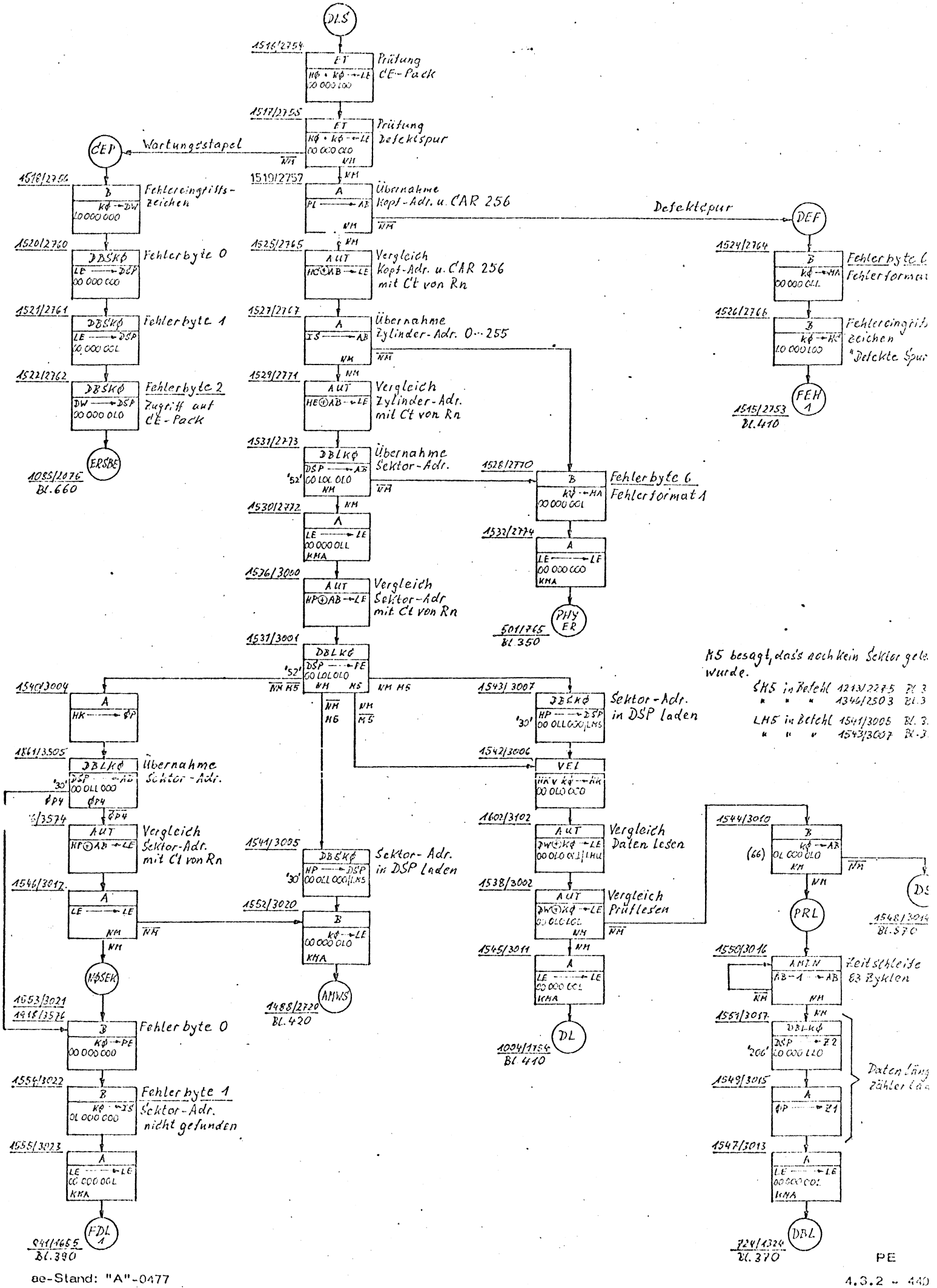


PE

4.3.2 - 43

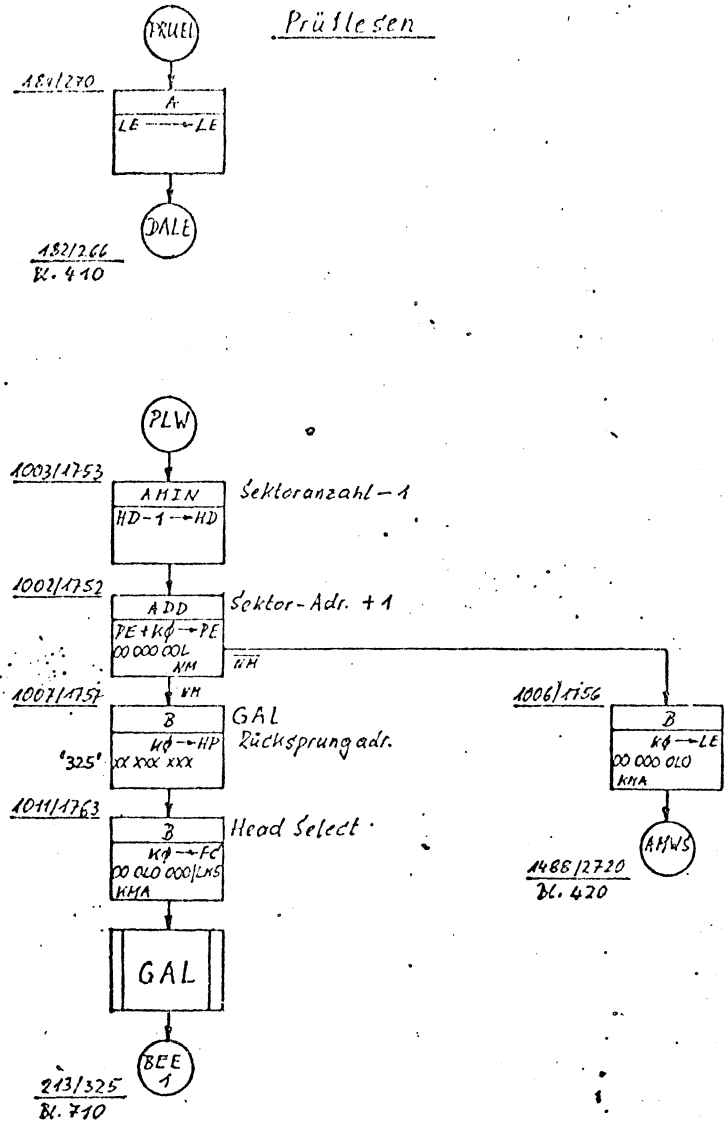
Bestell-Nr.: "A"-0477


DI.F 3
1501/2735
B
KØ→FC
00 010 000
Head Select
USH6
1494/2726
B
KØ→FC
00 000 000
ØP7
ØP7
Vor Head Advance muss Head Select weggenommen werden d.h. 12 Sektoren/Spur-Modus
1508/2744
AUT
PE⊕KØ→LE
00 001 101
Sektor-Adr. = 13?
1509/2745
B
KØ→FT
00 001 011
NM
TAG 11
1346/2502
B
KØ→IS
10 000 000
Fehlerbyte 1
"Unvollständige Spur im 12 Sektor/Spur-Modus"
1347/2503
B
KØ→FC
00 001 000/545
Head Advance
1128/2150
B
KØ→MA
00 000 000
Fehlerbyte 6
Fehlerformat 0
B
KØ→HP
00 000 011
(3)
1129/2151
B
KØ→HC
11 000 000
Fehlereingriffs-zeichen
1446/2646
AMIN
HP-1→HP
NM
Zeitschleife 4 Zyklen
FEH 2
1513/2751
Bl. 410
1447/2647
B
KØ→FT
00 001 011
TAG 11
1445/2645
B
KØ→FC
00 000 000
Head Advance
1311/2437
B
KØ→HP
00 001 001
(9)
1406/2576
AMIN
HP-1→HP
NM
Zeitschleife 10 Zyklen
1407/2577
ET
FS·KØ→LE
00 000 010
Prüfung End of Cylinder
1319/2447
ADD
PL+KØ→PL
00 000 001/LM
Kopf-Adr. +1
1502/2736
B
KØ→FT
01 001 001
TAG 9
1503/2737
B
KØ→FE
00 000 001/LM
Sektor-Adr. = 1
1255/2347
B
KØ→FC
00 000 010
Control Reset
Ist nötig, da "End of Cylinder" erst gemeldet wird, wenn bereits ein Kopf > 18 ausgewählt ist; dann aber auch Device Check.
1488/2727
ET
HK·KØ→HK
11 101 111
Merkbit löschen für "Einen gesuchten Sektor gefunden"
1523/2763
B
KØ→PE
00 010 000
Fehlerbyte 0
Zylinderende
DICH
1477/2705
Bl. 410
FDL 2
1285/2410
Bl. 710



ae-Stand: "A"-0477

PE


PRUEL
Prüflesen
Belegte Register bei Einsprung: LG: 4. SIZ (OP-Code)
HK: Weichenbyte 1
HD: Sektoranzahl
PL, PG: Zylinder-Adresse
HE: Kopf-Adresse
PE: Sektor-Adresse
1841/2270
A
LE ←— LE
DALE
1821/2266
Bl. 410
PLW
1003/1753
AMIN
HD-1 → HD
Sektoranzahl – 1
1002/1752
ADD
PE+KØ → PE
00 000 COL
NM
Sektor-Adr. + 1
ÜH
1007/1757
NH
B
KØ → HP
XX XXX XXX
'325'
GAL
Rücksprung adr.
1011/1763
B
KØ → FC
00 OLO 000 LMS
KHA
Head Select
GAL
BEE 1
213/3325
Bl. 710
1006/1756
B
KØ → LE
00 000 OLO
KHA
AMWS
1488/2720
Bl. 420

Sektor RO Schreiben

Belegte Register bei Einsprung: LG: 4. SIZ (ΔP-Code)
HK: Weichen byte 4
PL, ZS: Zylinder-Adresse
HE: Kopf-Adresse

SCYC

1281/1274 — A — LESCA Rücksprungadr. — '237' XX XXXXXX

LESCA

1415/1637 — A — LE → LE, 00 000 010, KMA

1260/2354 — B — TAG 5 Modul-Adr. durchschalten — KØ → FT, 10 000 101

1261/2355 — B — KØ → 21, 00 000 101 — Daten längen-zähler laden

1262/2356 — B — KØ → 22, 00 000 000

1257 — B — GALS Rücksprungadr. — KØ → HP, '242' XX XXX XXX

1271/2367 — A — Übernahme 316 Code — FS → HA, 00 000 001, KMA

GALS

1274/1247 — A — LE → LE, 00 000 010, KMA

1264/2360 — B — KØ → ST, 00 010 000

1265/2361 — B — KØ → DR, 00 000 000 — Pseudo Schreiben

1274/2372 — B — KØ → ST, 00 000 010

1275/2373 — B — USH 5 Rücksprungadr. — KØ → HP, '365' XX XXX XXX

1266/2362 — B — TAG 11 Fehler durchschalten — KØ → FT, 01 001 011, FLMU

1267/2363 — B — Head Select — KØ → FC, 00 010 000

USH5

15 — ET — Prüfung Write Inhibited — FS · KØ → LE, 00 01 0 000

1273/2371 — A — LE → LE, NM — NM → SSL

SSL — 1296/2420 Bl. 530

1297/2421 — P — ZHFS Rücksprungadr. — KØ → HP, '12' XXXXX XXX

ZHFS

1290/2412 — A — LE → LE, 00 000 001, FLMU, KMA

680/1250 — B — KØ → HP, 00 010 010, FLMU, MI

MI

683/1253 — AUT — HP ⊕ KØ → LE, 00 010 001 — 780/1414

782/1416 — AMIN — HP − 1 → HP, NM — Zeitschleife 4 Zyklen nötig, da der Index ca. 2 µs lang ist und bei Index-Write Gerät Fehler meldet.

781/1415 — B — TAG 11 Bus-In durchschalten — KØ → FT, 00 001 011, FLMU

684/1254 — B — Write Head Select — KØ → FC, 00 010 100, FLMU

840/1510 — AMIN — Zeitschleife 17 Zyklen — HP − 1 → HP, NM

841/1511 — AUT — Prüfung Write Sense Rd./Wr. Valid — FS ⊕ KØ → LE, 00 001 001

843/1513 — B — KØ → ST, 00 000 000, NM

687/1257 — B — KØ → HP, 01 001 111

696/1270 — A — HP → LE

692/1264 — MD MU / HD MU

80 Bytes 00...00 (G1)

694/1266 — AMIN — HP − 1 → HP, LND, NM

697/1271 — B — KØ → DR, 01 011 011 — 700/1274 — 1 Byte '00'

Übergabe Sync.-Byte (G1)

703/1276 — A — LE → LE, LND

SFA — 704/1300 Bl. 470

682/1252 — A — LE → LE, NM

681/1251 — A — LE → LE, 00 000 000, FLMU, KMA

IFE — 372/564 Bl. 340

684/1256 — B — Head Select — KØ → FC, 00 010 000

570/1072 — B — Sprungadr. für MWR — KØ → HP, '343' XX XXX XXX

571/1073 — A — HP → LE, 00 010 101, KMA, AMA

KWR — 483/743 Bl. 340

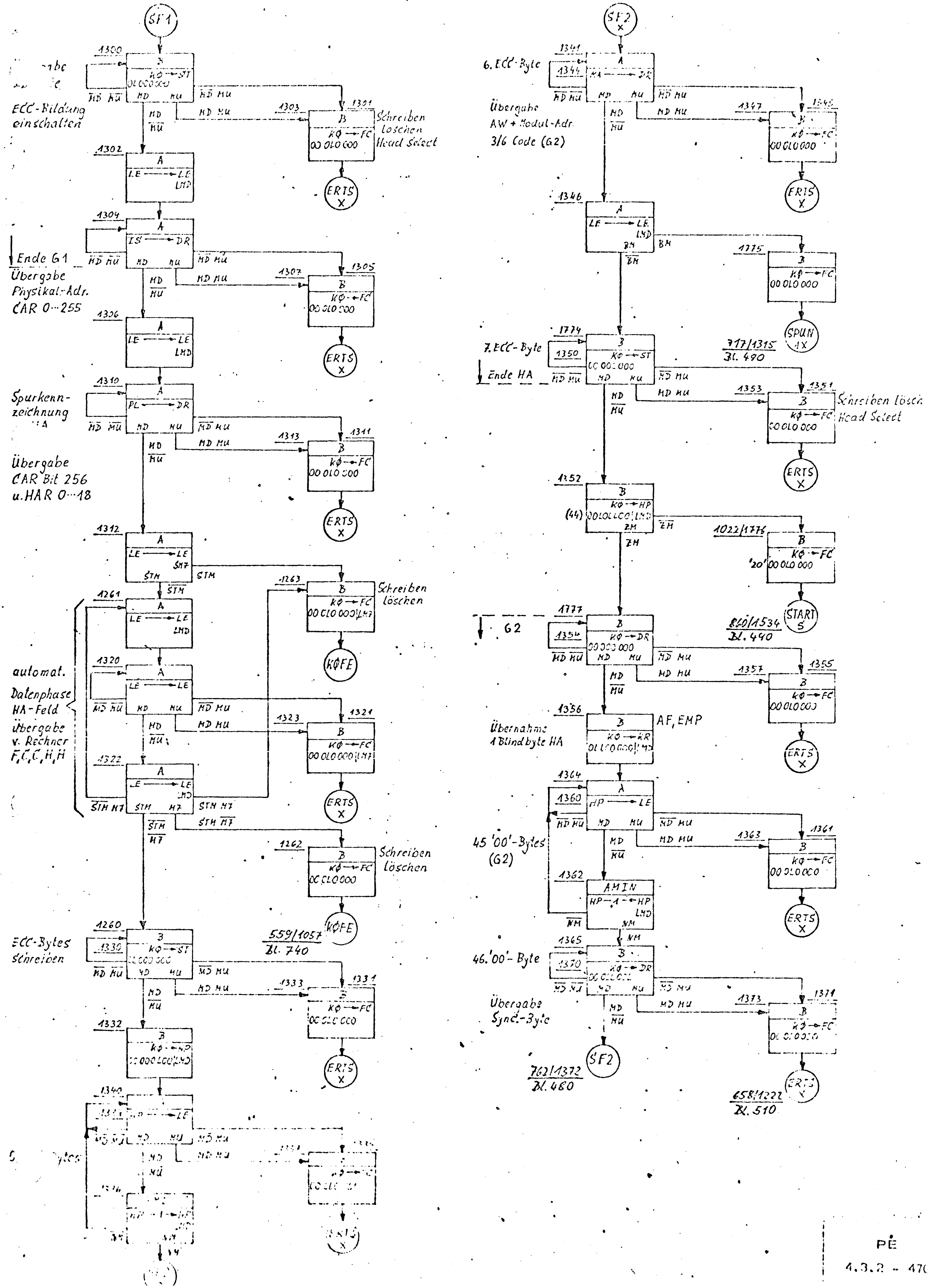
695/1267 — B — Head Select — KØ → FC, 00 010 000

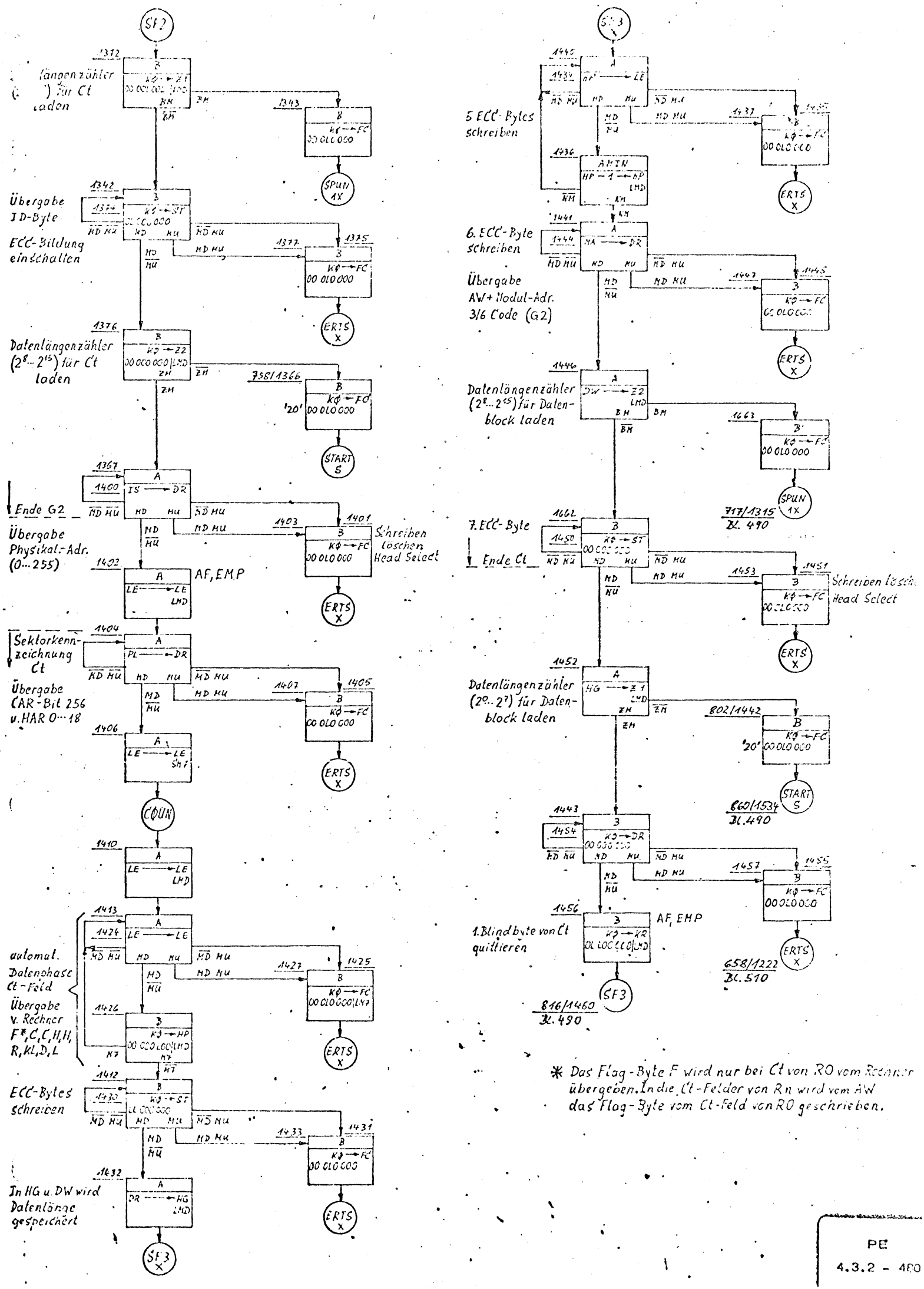
ERTS X

703/1277 — B — Head Select — KØ → FC, 00 010 000

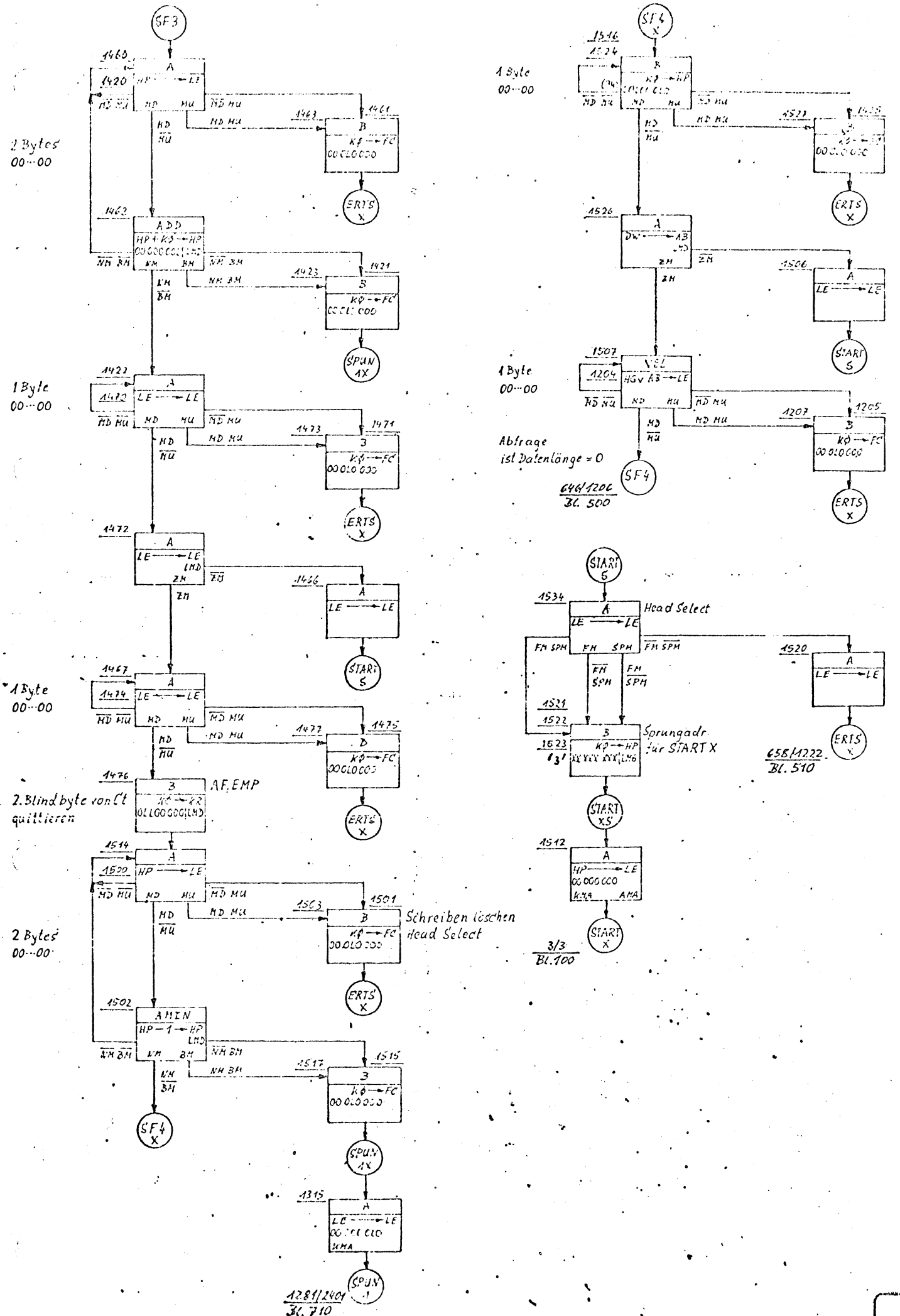
ERTS X — 658/1222 Bl. 510

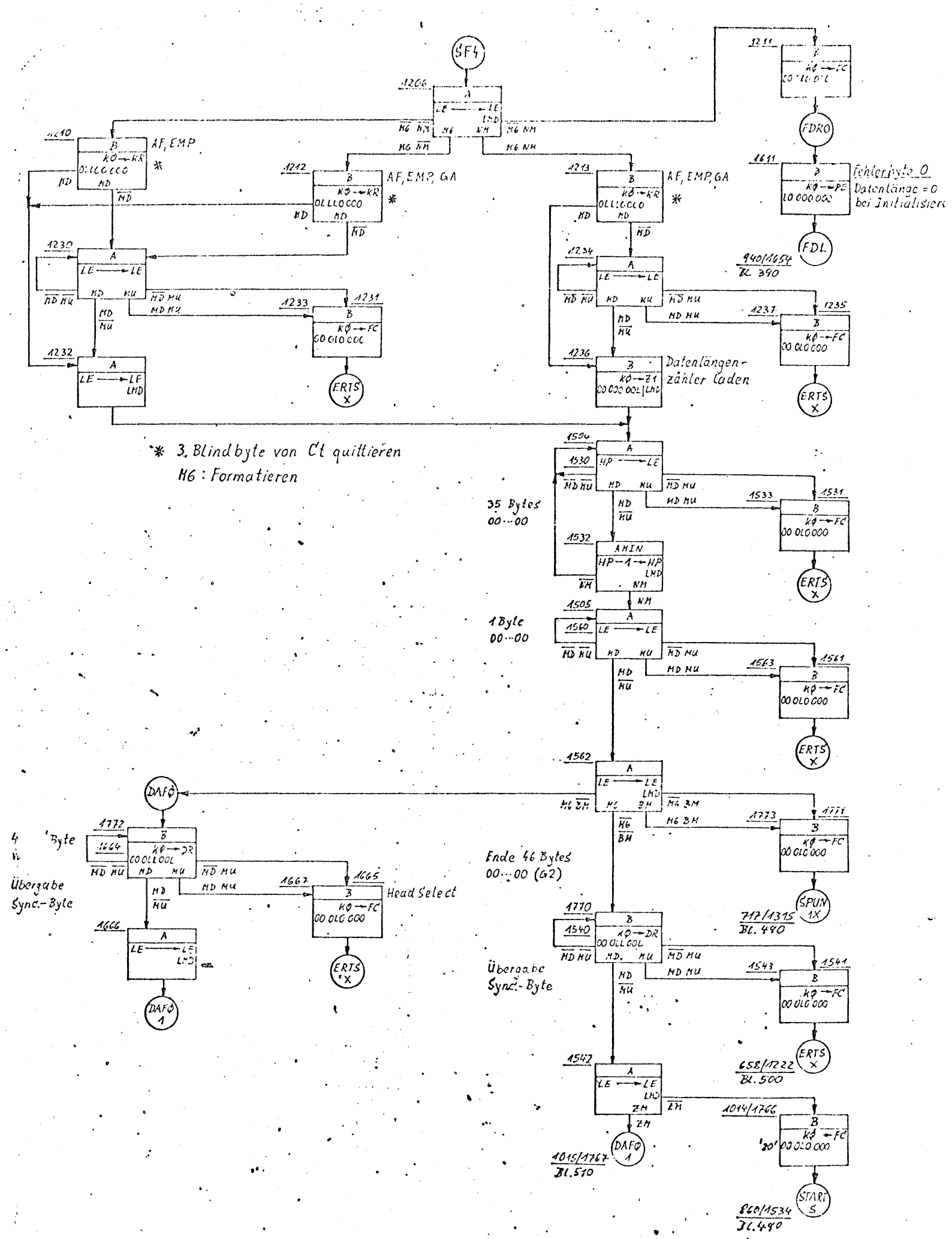
PE

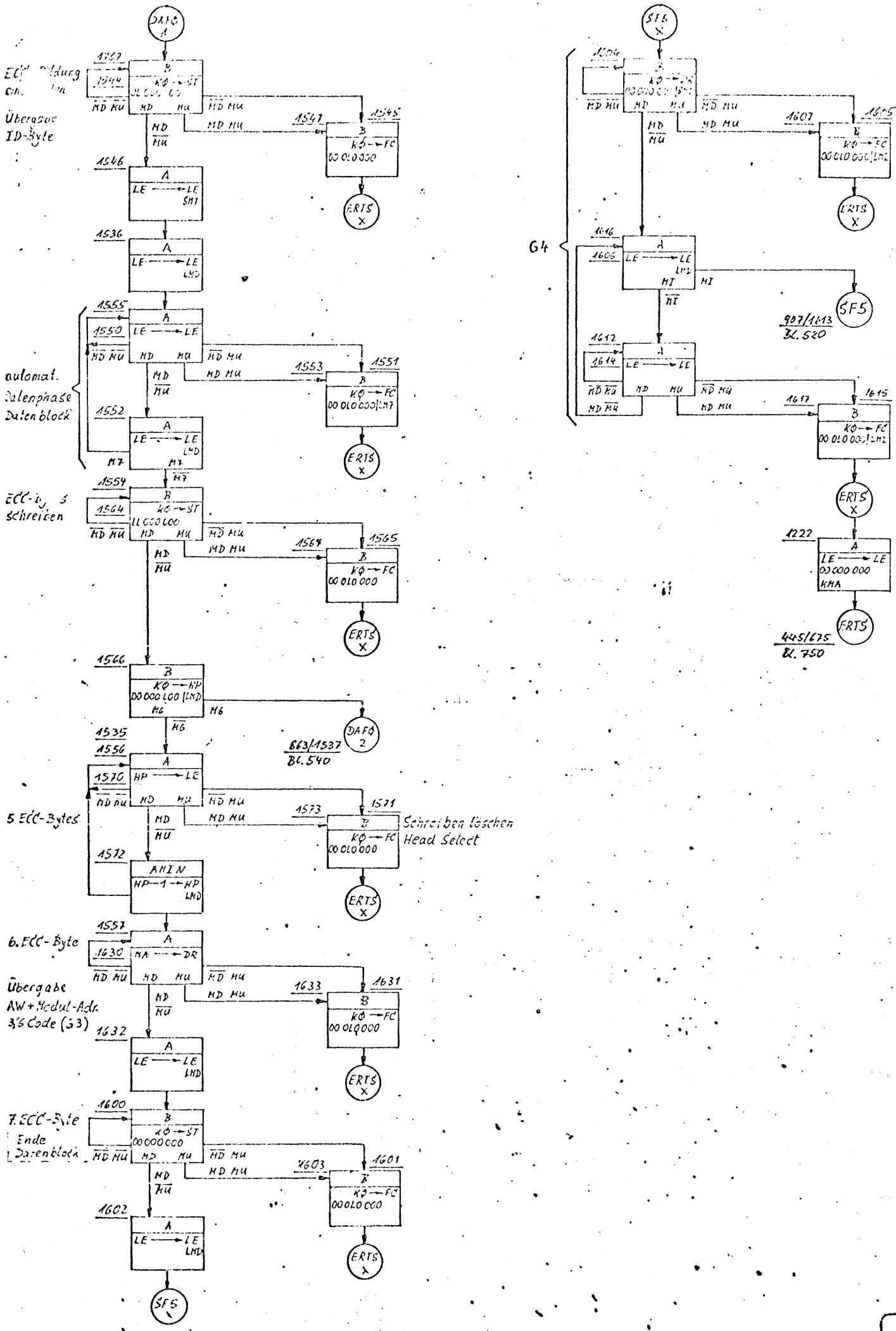

SF1
1300
ECC-Bildung einschalten
KØ → ST
01 000 000
1301
Schreiben löschen Head Select
KØ → FC
00 010 000
ERTS
1302
LE → LE
1304
IS → DR
Ende G1
Übergabe Physikal.-Adr. CAR 0…255
1305
1306
1310
PL → DR
Spurkennzeichnung
1311
Übergabe CAR Bit 256 u. HAR 0…18
1312
1263
Schreiben löschen
KØFE
1261
1320
automat. Datenphase HA-Feld Übergabe v. Rechner F, C, C, H, H
1321
1322
1262
Schreiben löschen
1260
ECC-Bytes schreiben
559/1057
Bl. 740
1331
1332
1340
SF2
1341
6. ECC-Byte
Übergabe AW + Modul-Adr. 3/6 Code (G2)
1345
1346
1775
SPUN
1774
7. ECC-Byte
Ende HA
717/1315
Bl. 490
1351
Schreiben löschen Head Select
1352
KØ → HP
(44)
1022/1776
'20'
START
1777
G2
860/1534
Bl. 490
1355
1356
AF, EMP
Übernahme 1 Blindbyte HA
1364
HP → LE
45 '00'-Bytes (G2)
1361
1362
AMIN
HP − 1 → HP
1365
46. '00'-Byte
Übergabe Sync-Byte
1371
762/1372
Bl. 480
658/1222
Bl. 510

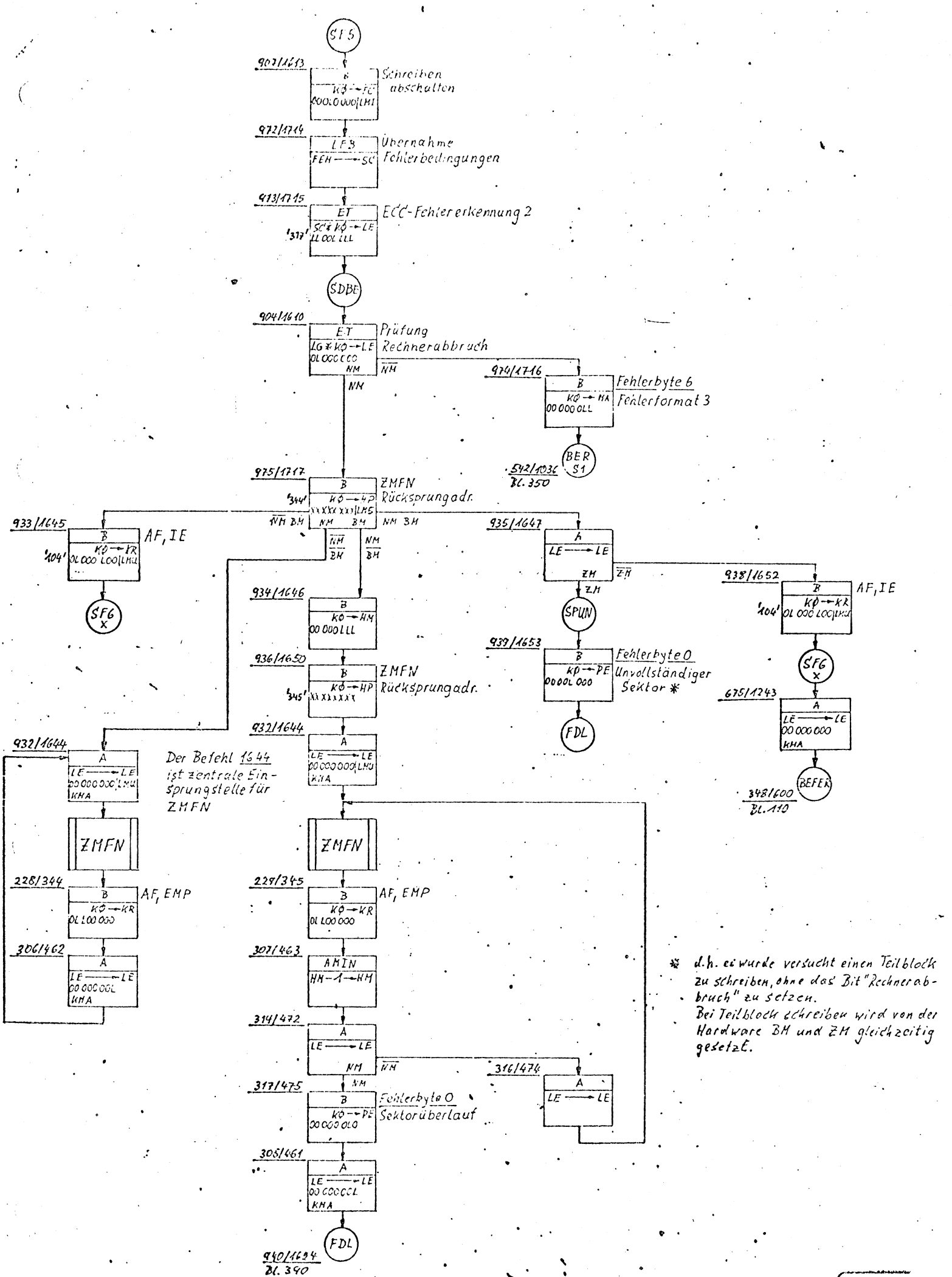


* Das Flag-Byte F wird nur bei Ct von R0 vom Rechner übergeben. In die Ct-Felder von Rn wird vom AW das Flag-Byte vom Ct-Feld von R0 geschrieben.


SF3
1460
A
1420
HP → LE
MD MU
MD MU
2 Bytes
00···00
1463
1461
B
KØ → FC
00 CL 0 000
ERTS X
1462
ADD
HP + K3 → HP
00 000 001
MM BM
MM BM
1423
1421
B
KØ → FC
00 CL 000
SPUN AX
1422
1 Byte
00···00
1472
LE → LE
1473
1471
3
KØ → FC
00 0L0 000
1466
START S
1467
1474
1477
1475
D
KØ → FC
00 0L0 00 3
1476
3
AF, EMP
2. Blindbyte von Ct quittieren
1514
1520
1503
1501
Schreiben löschen
Head Select
2 Bytes
00···00
1502
AMIN
HP − 1 → HP
LMD
1517
1515
SF4 X
1315
SPUN A1
12 81/2401
Bl. 710
1516
1524
1 Byte
00···00
1521
1526
A
DW → AB
ZM
1506
START S
1507
1204
VEL
HGV K3 → LE
1207
1205
Abfrage
ist Datenlänge = 0
646/1206
Bl. 500
START S
1534
Head Select
FM SPM
1520
1521
1522
3
Sprungadr. für START X
1523
13
658/1222
Bl. 510
START XS
1512
HP → LE
00 000 000
KMA AMA
3/3
Bl. 100


SF4
1206
1210
AF, EMP
1212
AF, EMP, GA
1213
AF, EMP, GA
1211
FDRO
1611
Fehlerbyte 0
Datenlänge = 0
bei Initialisierung
FDL
940/1654
Bl. 390
1230
1231
1232
1233
1234
1235
1236
Datenlängen-
zähler laden
1237
ERTS
* 3. Blindbyte von Ct quittieren
M6 : Formatieren
1504
1530
1531
1532
1533
35 Bytes
00...00
1505
1540
1541
1543
1 Byte
00...00
1561
1562
1563
1771
1772
1773
Ende 46 Bytes
00...00 (G2)
1664
1665
1666
1667
Head Select
DAFØ
1
1770
SPUN
1X
717/1315
Bl. 490
Übergabe
Sync.-Byte
1542
658/1222
Bl. 500
1014/1766
'20'
1015/1767
Bl. 510
START
S
860/1534
Bl. 490

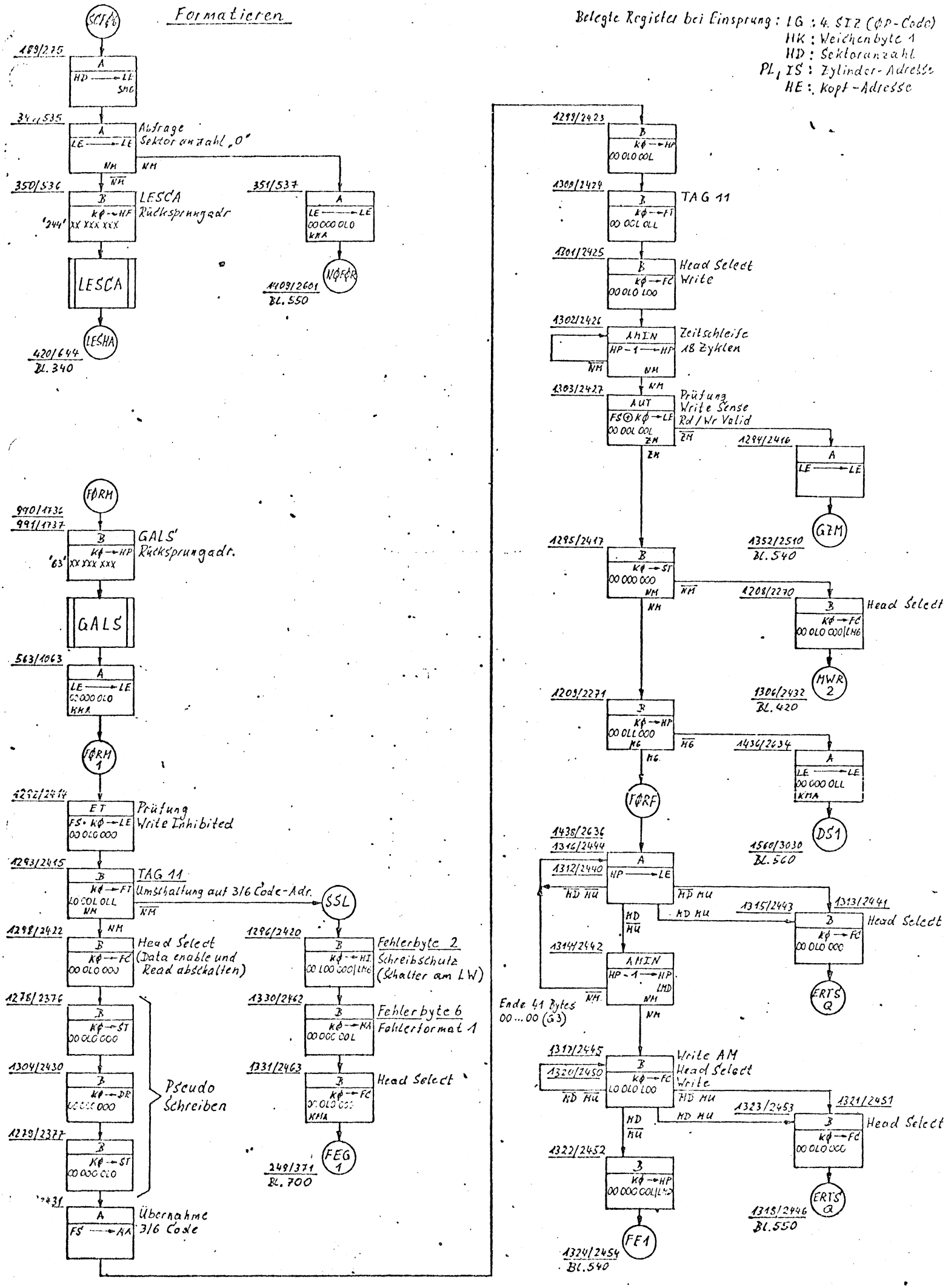

DAFÜ 1
1547
1544
B
KØ → ST
MD MU
MD MU
MD MU
1547
1545
B
KØ → FC
00 010 000
ERTS X
1546
A
LE → LE
SM1
1536
A
LE → LE
LM0
1555
A
1550
LE → LE
MD MU
MD MU
MD MU
1553
1551
B
KØ → FC
00 010 000 LM1
1552
A
LE → LE
LM0
M7
M7
1554
B
1564
KØ → ST
11 000 000
MD MU
MD MU
MD MU
1567
1565
B
KØ → FC
00 010 000
ERTS X
1566
B
KØ → HP
00 000 100 LM0
M6
M6
DAFÜ 2
1535
1556
A
1570
HP → LE
MD MU
MD MU
MD MU
1573
1571
B
KØ → FC
00 010 000
Schreiben löschen
Head Select
ERTS X
1572
AMIN
HP − 1 → HP
LM0
1557
A
1630
MA → DR
MD MU
MD MU
MD MU
1633
1631
B
KØ → FC
00 010 000
ERTS X
1632
A
LE → LE
LM0
1600
B
KØ → ST
00 000 000
MD MU
MD MU
MD MU
1603
1601
B
KØ → FC
00 010 000
ERTS X
1602
A
LE → LE
LM0
SF5 X
SFÜ X
1604
B
KØ → ST
MD MU
MD MU
MD MU
1607
1605
B
KØ → FC
00 010 000 LM2
ERTS X
1616
A
1606
LE → LE
LM2
MI
MI
MI
SF5
1612
A
1614
LE → LE
MD MU
MD MU
MD MU
1617
1615
B
KØ → FC
00 010 000 LM2
ERTS X
1222
A
LE → LE
00 000 000
KMA
FRTS
G4
ECC-Bildung
Übergabe
ID-Byte
automat.
Datenphase
Datenblock
ECC-Byte
schreiben
5 ECC-Bytes
6. ECC-Byte
Übergabe
AW + Modul-Adr.
3/6 Code (33)
7. ECC-Byte
Ende
Datenblock
907/1613
Bl. 520
445/675
Bl. 750
663/1537
Bl. 540

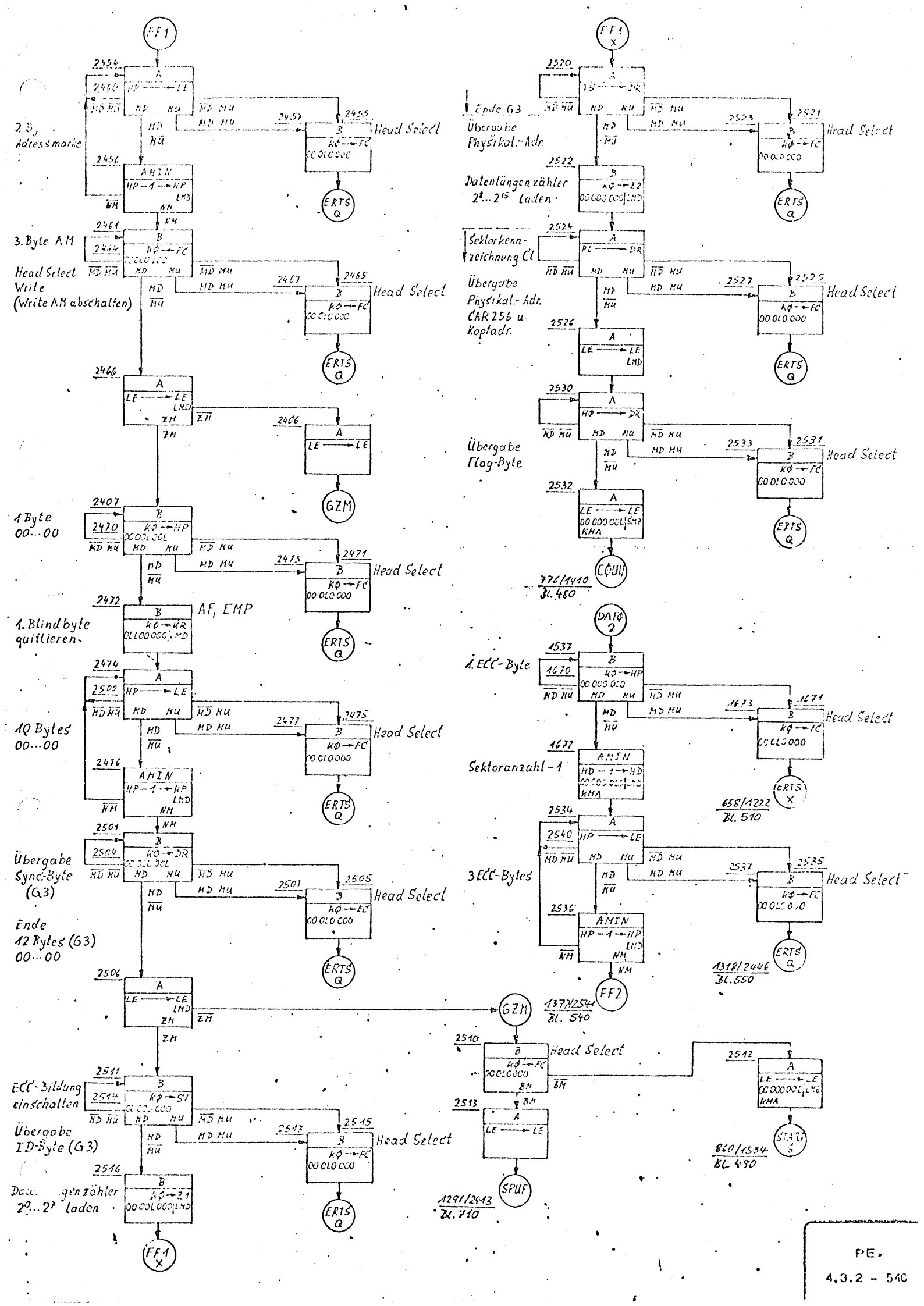


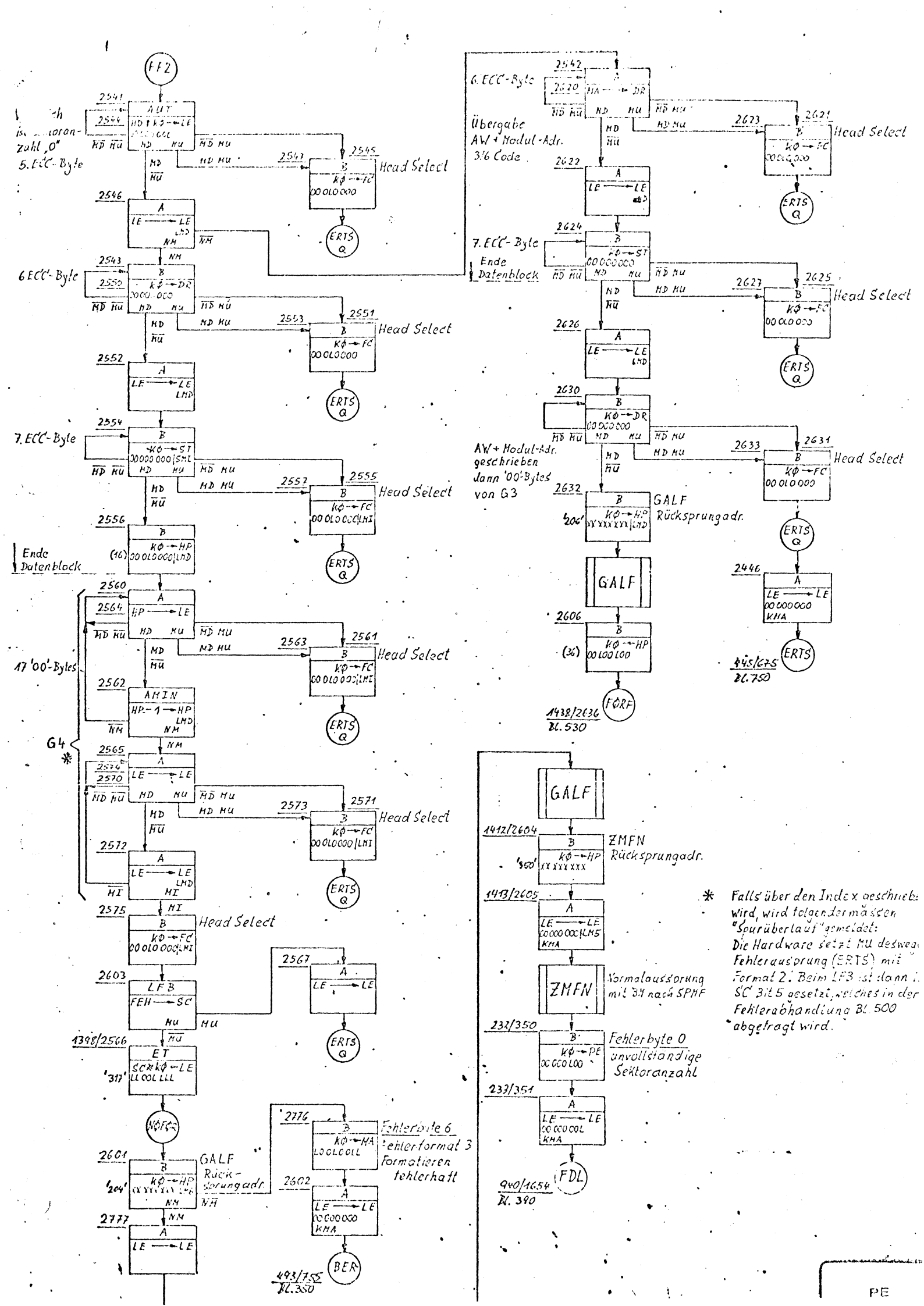
* d.h. es wurde versucht einen Teilblock zu schreiben, ohne das Bit "Rechnerabbruch" zu setzen.
Bei Teilblock schreiben wird von der Hardware BM und ZM gleichzeitig gesetzt.

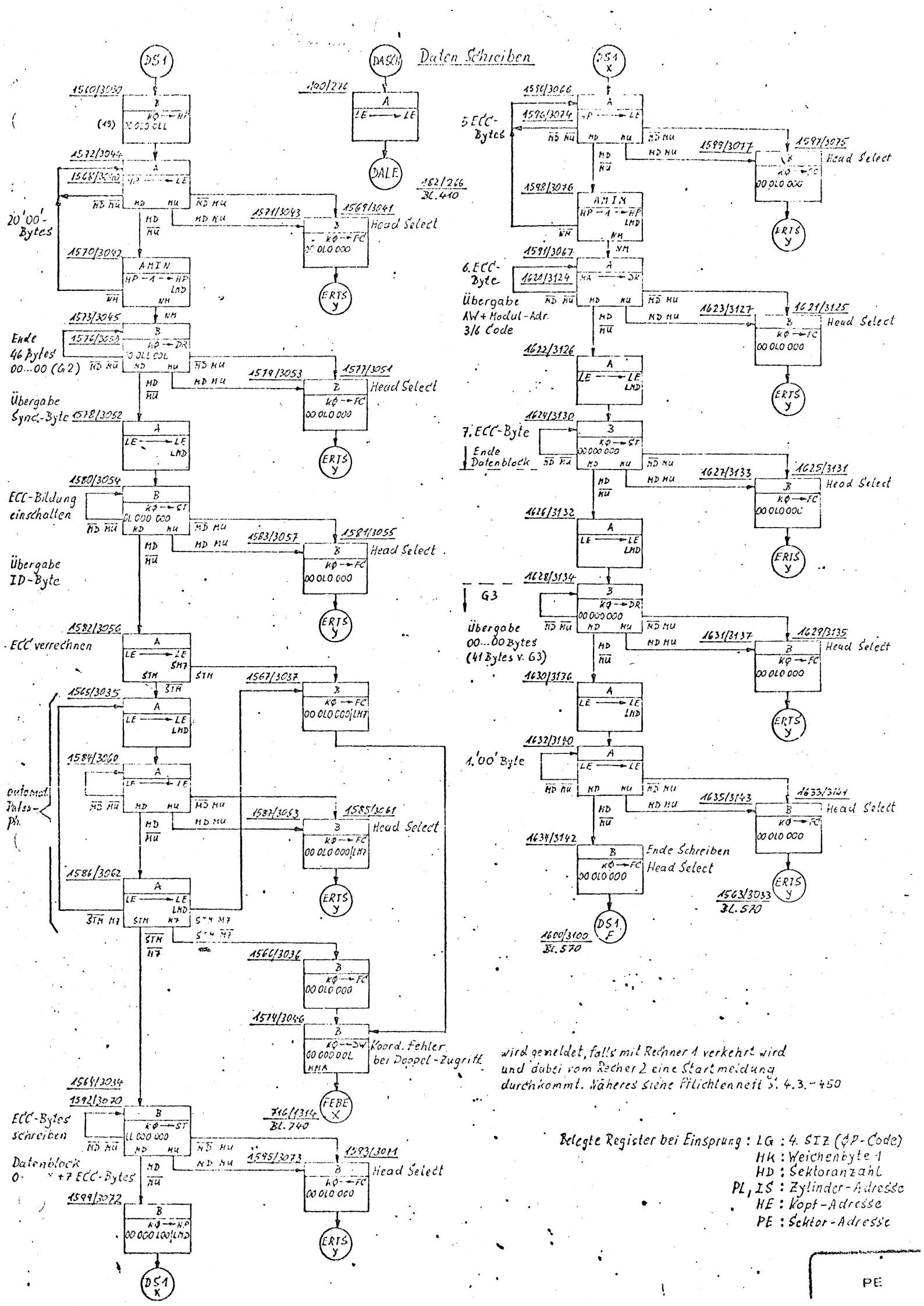
Formatieren

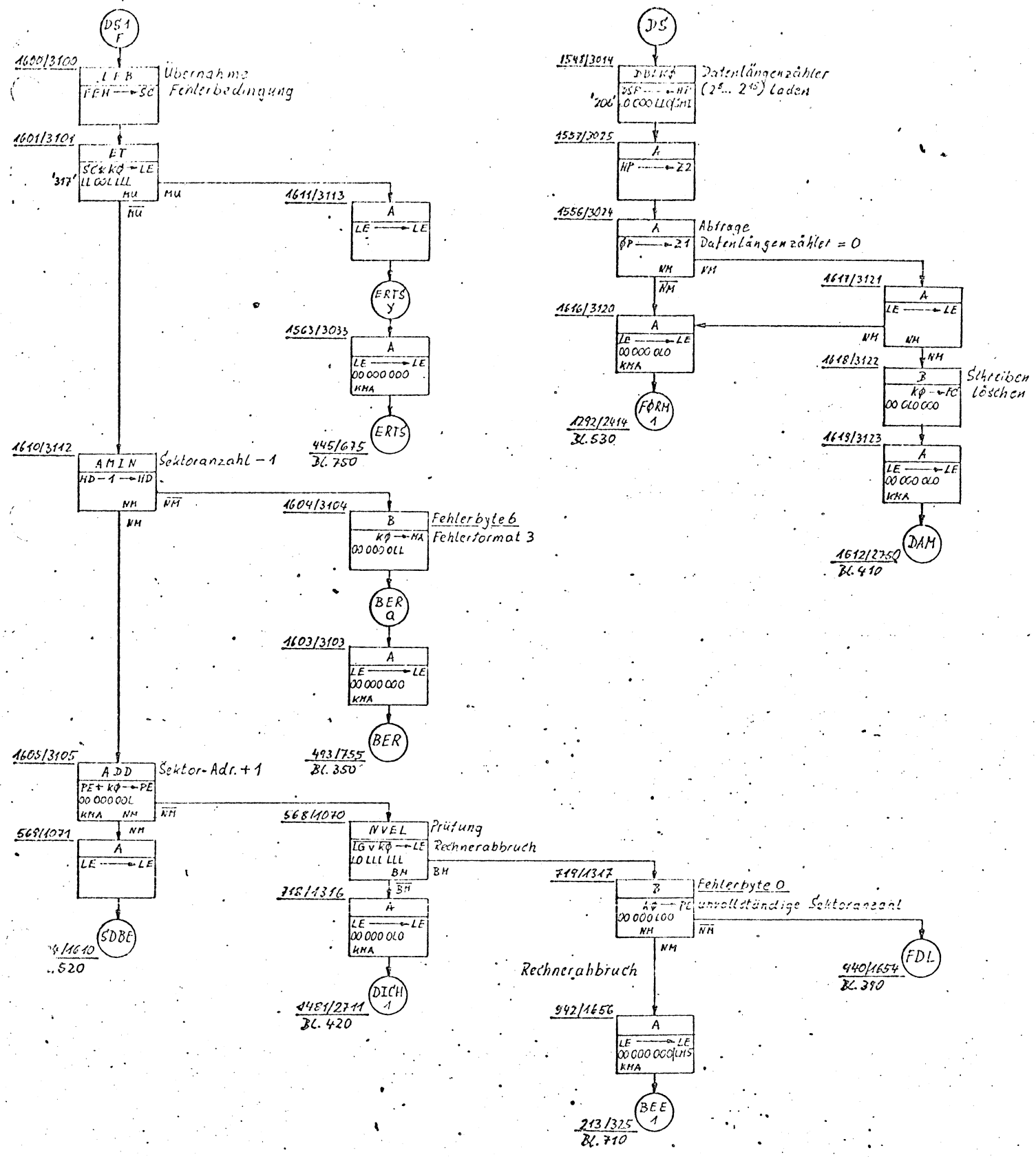
Belegte Register bei Einsprung: LG : 4. STZ (ØP-Code)
HK : Weichenbyte 1
HD : Sektoranzahl
PL, IS : Zylinder-Adresse
HE : Kopf-Adresse




FF1
2454
2460
HP → LE
MD MU
2455
Head Select
2457
2456
AMIN
HP − 1 → HP
LMD
NM
ERTS Q
2 Bytes Adressmarke
2461
3. Byte AM
2464
Head Select Write (Write AM abschalten)
2465
2467
2466
LE → LE
ZM
2406
GZM
2407
1 Byte 00...00
2470
2471
2473
2472
1. Blindbyte quittieren
AF, EMP
2474
2500
2475
2477
10 Bytes 00...00
2476
2501
2504
Übergabe Sync-Byte (G3)
2505
2501
Ende 12 Bytes (G3) 00...00
2506
2511
ECC-Bildung einschalten
2514
2512
Übergabe ID-Byte (G3)
2515
2516
Datenlängenzähler 2⁰...2⁷ laden
FF1 X
2520
Ende G3 Übergabe Physikal.-Adr.
2521
2523
2522
Datenlängenzähler 2⁸...2¹⁵ laden
2524
Sektorkennzeichnung C1
2525
2527
Übergabe Physikal.-Adr. CAR 256 u. Kopfadr.
2526
2530
Übergabe Flag-Byte
2531
2533
2532
CCUK
776/1410
Bl. 480
DATÜ 2
1537
1. ECC-Byte
1670
1671
1673
1672
Sektoranzahl − 1
658/1222
Bl. 510
2534
2540
3 ECC-Bytes
2535
2537
2536
FF2
1377/2544
Bl. 540
2510
2513
SPUF
1291/2413
Bl. 710
1318/2446
Bl. 550
860/1534
Bl. 490
START S


FF2
2541
AUT
MD MU → LE
2544
MD MU
2545
B
KØ → FC
00 010 000
Head Select
ERTS Q
2546
A
LE → LE
LMD
NM
2543
B
KØ → DR
2550
00 000 000
2551
2553
2552
2554
B
KØ → ST
00 000 000 SML
2555
2557
2556
B
KØ → HP
(16)
00 010 000 LMD
Ende Datenblock
2560
A
HP → LE
2564
2561
2563
2562
AMIN
HP-1 → HP
LMD
NM
G4 *
2565
2574
2570
2571
2573
2572
HI
2575
2603
LFB
FEH → SC
2567
1398/2566
ET
SC KØ → LE
'317'
NØFC
2601
GALF
Rück-sprungadr.
'204'
2777
2776
Fehlerbyte 6
Fehlerformat 3
formatieren fehlerhaft
2602
BER
493/755
Bl. 350
6. ECC-Byte
2542
2620
MA → DR
Übergabe
AW + Modul-Adr.
3/6 Code
2623
2621
2622
LE → LE
7. ECC-Byte
Ende Datenblock
2624
KØ → ST
00 000 000
2627
2625
2626
2630
KØ → DR
00 000 000
AW + Modul-Adr. geschrieben
dann '00'-Bytes von G3
2633
2631
2632
GALF
Rücksprungadr.
'206'
GALF
2606
(36)
FØRF
1438/2636
Bl. 530
2446
KMA
ERTS
445/675
Bl. 750
GALF
1412/2604
ZMFN
Rücksprungadr.
'300'
1413/2605
LMS
ZMFN
Normalaussprung mit 3/4 nach SPMF
232/350
Fehlerbyte 0
unvollständige Sektoranzahl
233/351
FDL
940/1654
Bl. 390
* Falls über den Index geschrieben wird, wird folgendermassen "Spurüberlauf" gemeldet: Die Hardware setzt MU deswegen Fehlerausgang (ERTS) mit Format 2. Beim LFB ist dann SC Bit 5 gesetzt, welches in der Fehlerabhandlung Bl. 500 abgefragt wird.
PE


Daten Schreiben
DS1
DASCH
DALE
182/266
BL.410
100/276
1560/3040
1572/3044
1568/3040
1571/3043
1569/3041
Head Select
20 '00'-Bytes
1570/3042
1573/3045
1576/3050
Ende 46 Bytes 00...00 (G2)
1579/3053
1577/3051
Übergabe Sync.-Byte
1578/3052
1580/3054
ECC-Bildung einschalten
1583/3057
1581/3055
Übergabe ID-Byte
1582/3056
ECC verrechnen
1567/3037
1565/3035
1584/3060
automat. Talss-ph.
1587/3063
1585/3061
1586/3062
1566/3036
1574/3046
Koord. Fehler bei Doppel-Zugriff
1564/3034
1592/3070
ECC-Bytes schreiben
716/1314
BL.740
FEBE
1595/3073
1593/3071
Datenblock 0 ... +7 ECC-Bytes
1599/3072
DS1 X
1556/3066
1576/3074
5 ECC-Bytes
1599/3077
1597/3075
1598/3076
1591/3067
6. ECC-Byte
1621/3124
Übergabe AW + Modul-Adr. 3/6 Code
1623/3127
1621/3125
1622/3126
1624/3130
7. ECC-Byte
Ende Datenblock
1627/3133
1625/3131
1626/3132
1628/3134
G3
Übergabe 00...00 Bytes (41 Bytes v. G3)
1631/3137
1629/3135
1630/3136
1632/3140
1 '00' Byte
1635/3143
1633/3141
1634/3142
Ende Schreiben
Head Select
1563/3033
BL.570
1600/3100
BL.570
wird gemeldet, falls mit Rechner 1 verkehrt wird und dabei vom Rechner 2 eine Startmeldung durchkommt. Näheres siehe Pflichtenheft S. 4.3.-450
Belegte Register bei Einsprung : LG : 4. SIZ (ÜP-Code)
HK : Weichenbyte 1
HD : Sektoranzahl
PL, IS : Zylinder-Adresse
HE : Kopf-Adresse
PE : Sektor-Adresse
PE


DS1 F
1600/3100
LFB
FFH → SC
Übernahme Fehlerbedingung
1601/3101
ET
SC & KØ → LE
LL ØØL LLL
'317'
HU
MU
1611/3113
A
LE → LE
ERTS Y
1563/3033
A
LE → LE
00 000 000
KHA
ERTS
445/6735
Bl. 750
1610/3112
AMIN
HD − 1 → HD
Sektoranzahl − 1
NM
1604/3104
B
KØ → HA
00 000 0LL
Fehlerbyte 6
Fehlerformat 3
BER Q
1603/3103
A
LE → LE
00 000 000
KHA
BER
493/1755
Bl. 350
1605/3105
ADD
PE + KØ → PE
00 000 00L
KHA
NM
Sektor-Adr. + 1
569/1071
A
LE → LE
SDBE
4/1610
..520
568/1070
NVEL
LG v KØ → LE
LO LLL LLL
BH
Prüfung
Rechnerabbruch
318/1316
A
LE → LE
00 000 0L0
KHA
DICH 1
1481/2711
Bl. 420
DS
1545/3014
DBLKØ
DSP → HT
0 000 LLØ/1m1
'206'
Datenlängenzähler (2⁵ ... 2¹⁵) laden
1537/3025
A
HP → Z2
1556/3024
A
ØP → Z1
Abfrage Datenlängenzähler = 0
NM
1616/3120
A
LE → LE
00 000 0L0
KHA
FØRH 1
1292/2414
Bl. 530
1617/3121
A
LE → LE
NM
1618/3122
B
KØ → FC
00 0L0 000
Schreiben löschen
1619/3123
A
LE → LE
00 000 0L0
KHA
DAM
1612/2750
Bl. 410
719/1317
B
ÄØ → FC
00 000 L00
NM
Fehlerbyte 0
unvollständige Sektoranzahl
FDL
440/1654
Bl. 390
Rechnerabbruch
942/1656
A
LE → LE
00 000 000/LHS
KHA
BEE 1
213/1325
Bl. 710

ECCP

1636/3144 B — Read Select Save Sector

1637/3145 DBSKØ '31'

1638/3146 DBSKØ '32'

1639/3147 DBSKØ '33'

1640/3150 B — Save Sector wegnehmen

1641/3151 DBSKØ '34'

1642/3152 DBSKØ '35'

1643/3153 DBSKØ '36'

1644/3154 DBSKØ '37'

1645/3155 DBSKØ '40'

1646/3156 DBSKØ '41'

1647/3157 DBSKØ '42'

1648/3160 DBSKØ '43'

1649/3161 DBSKØ '44'

1650/3170 B — KØ für Zeitschleife ca. 16 µs zwischen Save Sector u. Read Sector Register

1651/3171 AMIN PE-1→PE

1652/3172 B

1658/3173 X — USH7 Rücksprungadr. '200'

1660/3174 J — TAG 5

1661/3175 B — Read Sector Reg.

USH7

1664/3200 AMIN — Der gelesene Winkel muss zum Wiederfinden um 1 vermindert werden

1655/3167 ET — "High Side of Sector" ausblenden

1650/3162 DBSKØ '50'

1665/3201 B — Vorbesetzung für ACCU '00 00 B0' PE HC HE (B,0)

1666/3202 B (23)

1667/3203 A

1668/3204 AMIN — d.h. P011...P021 = 0

1671/3207 SHPØ — Schiften FG FO

1672/3210 B (9,1)

1674/3212 B (8,D)

1675/3213 B (D,0)

1676/3214 B — ADDUP Rücksprungadr. '226'

ADDUP

1668/3226 B — ULAD + ADDUP Rücksprungadr. '203'

ULAD

Leistung: Es wird untersucht ob der Fehler korrigierbar ist. Wenn nicht, erfolgt Fehlermeldung. Wenn ja, wird bei HA u. Ct Fehler der Fehler korrigiert und hinter der fehlerhaften Stelle weitergelesen. Bei Daten-Fehler werden die Korrekturdaten ins DSP hinterlegt und Fehler gemeldet.

Versorgung: HI: Rücksprungadr.
DW: Merkbyte für HA/Ct/Daten-Korrekturversuch (DW = 1/2/0)

Sede7

P3 ≠ P0 *	1	B	KØ→HG	AF	Spr.→CØLI 2		1793/3401
	2	"	" HF	36	" CØLI 3		1794/3402
	3	"	" AB	A0	" ECØLI		1795/3403
P2 ≠ P0 *	4	"	" HG	B3	" CØLI 6		1796/3404
P2,P3 ≠ P0 *	5	"	" HG	E1	" CØLI 8		1797/3405
	6	"	" HF	09	" CØLI 7		1798/3406
	7	"	" AB	A0	" ECØLI		1799/3407
	8	"	" HF	2E	" CØLI 9		1800/3410
	9	"	" AB	60	" ECØLI		1801/3411
	10	"	" HF	55	" CØLI 11		1502/3412
	11	"	" AB	A0	" ECØLI		1803/3413
	12	"	" HF	8C	" CØLI 13		1804/3414
	13	"	" AB	40	" ECØLI		1805/3415
	14	"	" HF	5F	" CØLI 15		1806/3416
	15	"	" AB	40	" ECØLI		1807/3417
P1 ≠ P0 *	16	"	" HG	EE	" CØLI 10		1508/3420
P1,P3 ≠ P0 *	17	"	" HG	9D	" CØLI 12		1809/3421
	18	"	" HF	84	" CØLI 19		1810/3422
	19	"	" AB	00	" ECØLI		1511/3423
P1,P2 ≠ P0 *	20	"	" HG	A1	" CØLI 14		1512/3424
P1,P2,P3 ≠ P0 *	21	B	KØ→HG	DF	Spr.→CØLI 18		1813/3425

* Einsprungstellen

ECØLI

1700/3244 B — ADDUP Rücksprungadr.

ADDUP

1701/3245 B — ULAD u. ADDUP Rücksprungadr. '234'

ULAD

1670/3206 B (1,5) — Maskenvorbesetzung für Vergleich von P0 mit P1 ... P3 *

1669/3205 B (8,9)

1692/3234 A

1693/3235 ET HN · AB → HN

1694/3236 AMIN HP-1→HP

FEØR

1697/3241 BL. 600

1696/3240 SHP123 Shift von P1 P2 P3

1698/3242 A

CØLI

HN0 = L : P3 war noch nicht gleich
HN2 = L : P2 " " " "
HN4 = L : P1 " " " "
Tritt irgendwann Gleichheit auf, wird das entsprechende Masken bit gelöscht d.h. danach wird dieser Vergleich nicht mehr ausgewertet.

ERRØ

1699/3243
1673/3211
1705/3251
1346/3322

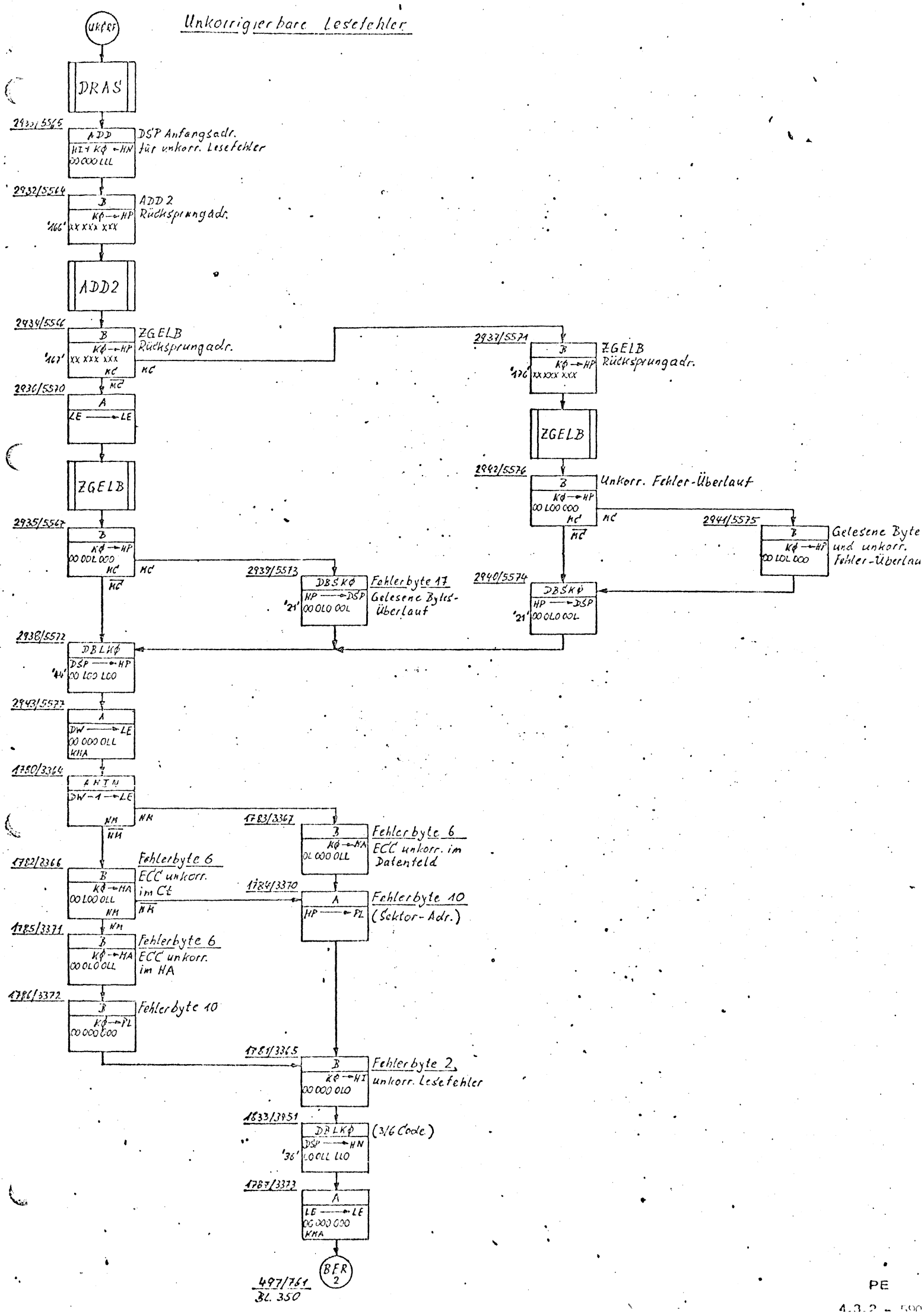
B — DRAS Rücksprungadr. '965'

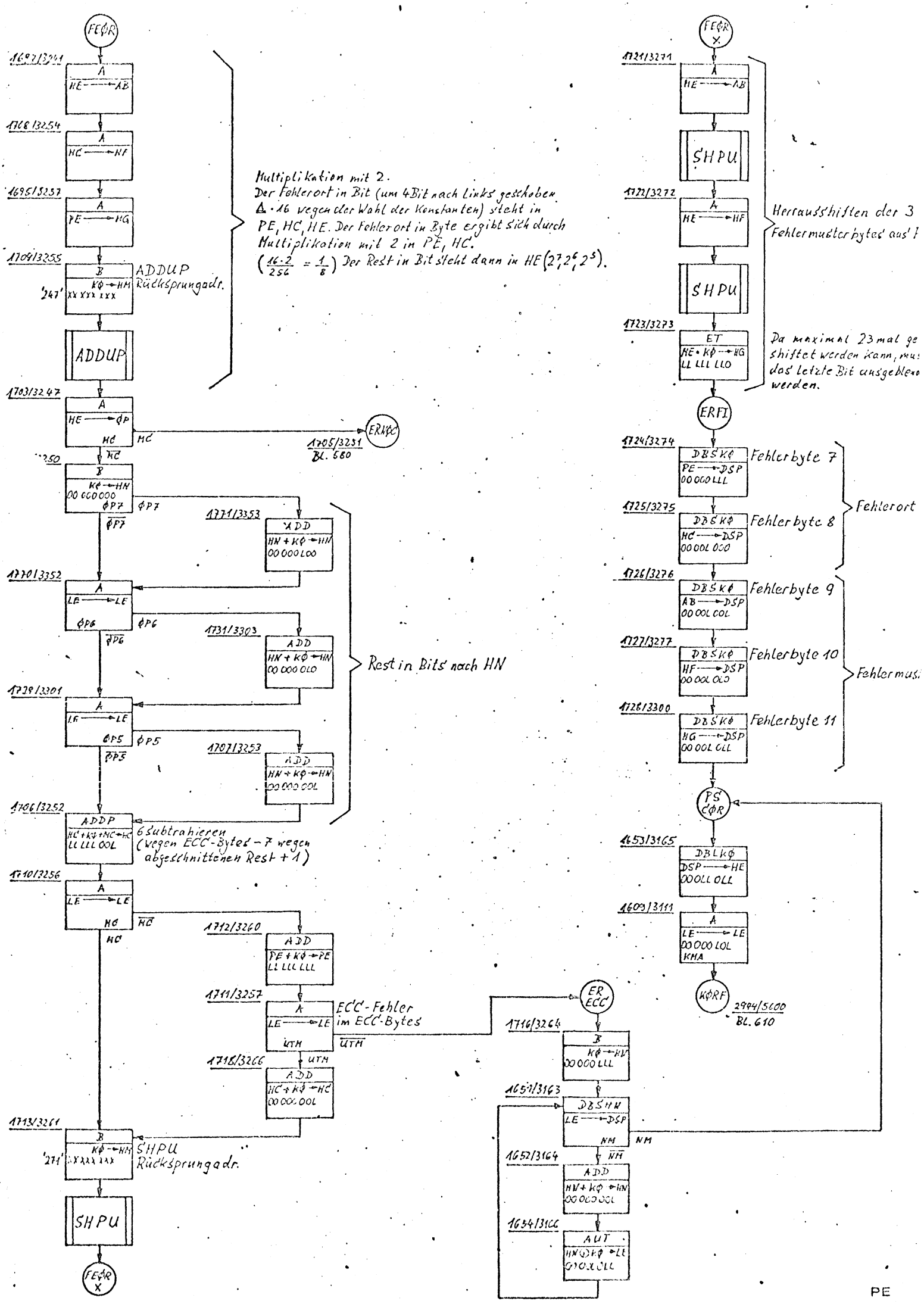
UKØRR

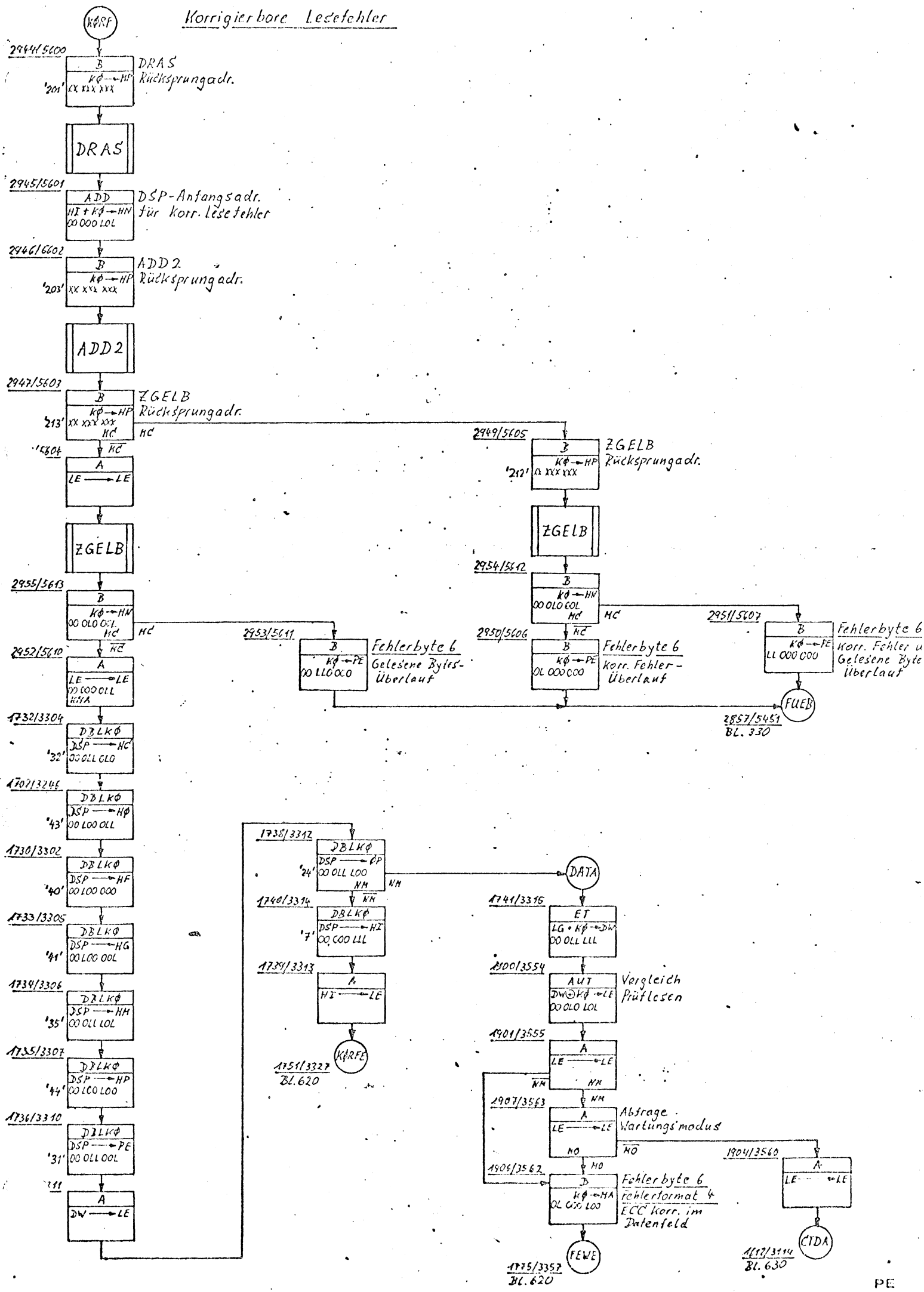
2820/5404
BL. 590

PE

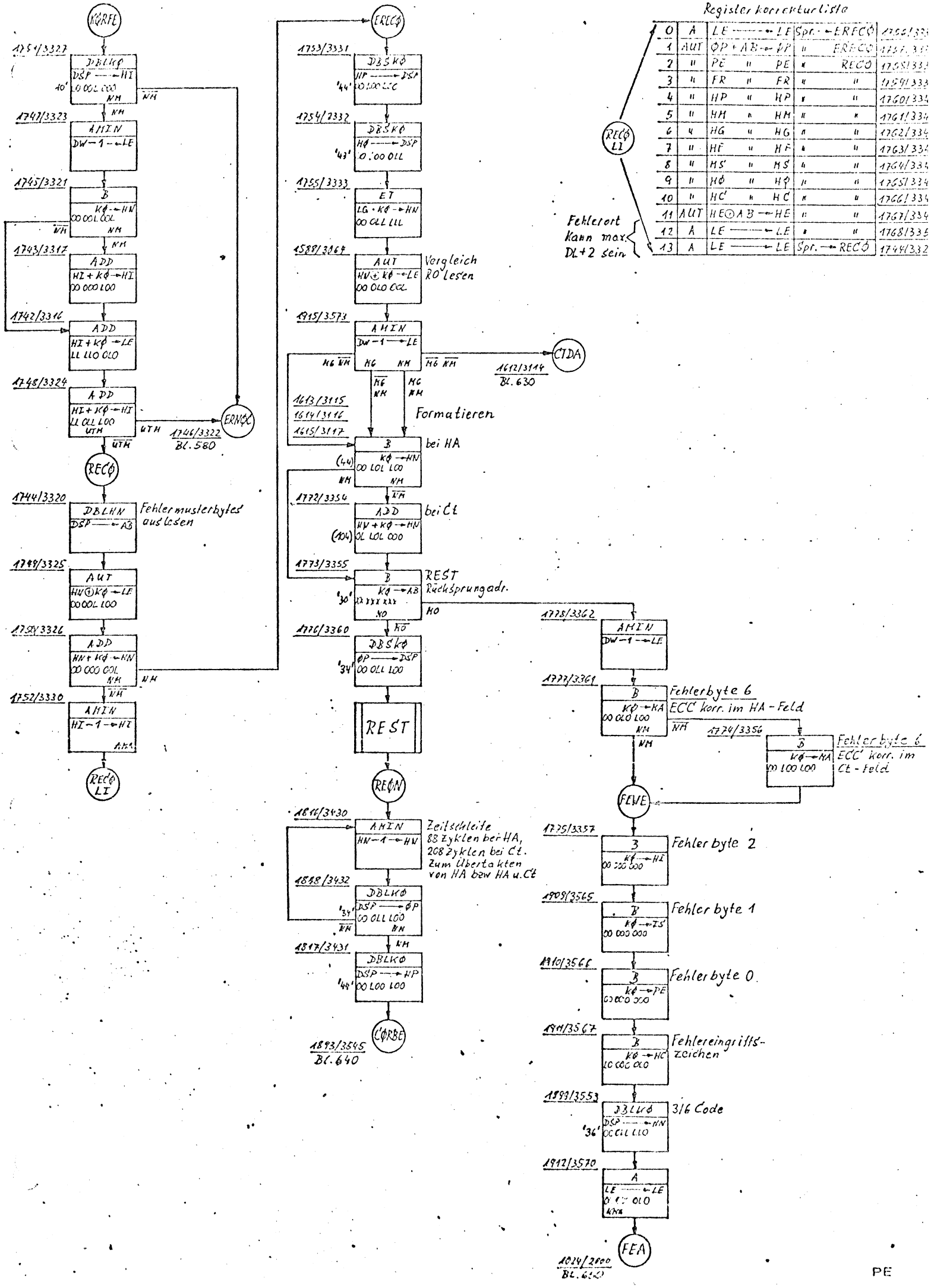
Unkorrigierbare Lesefehler



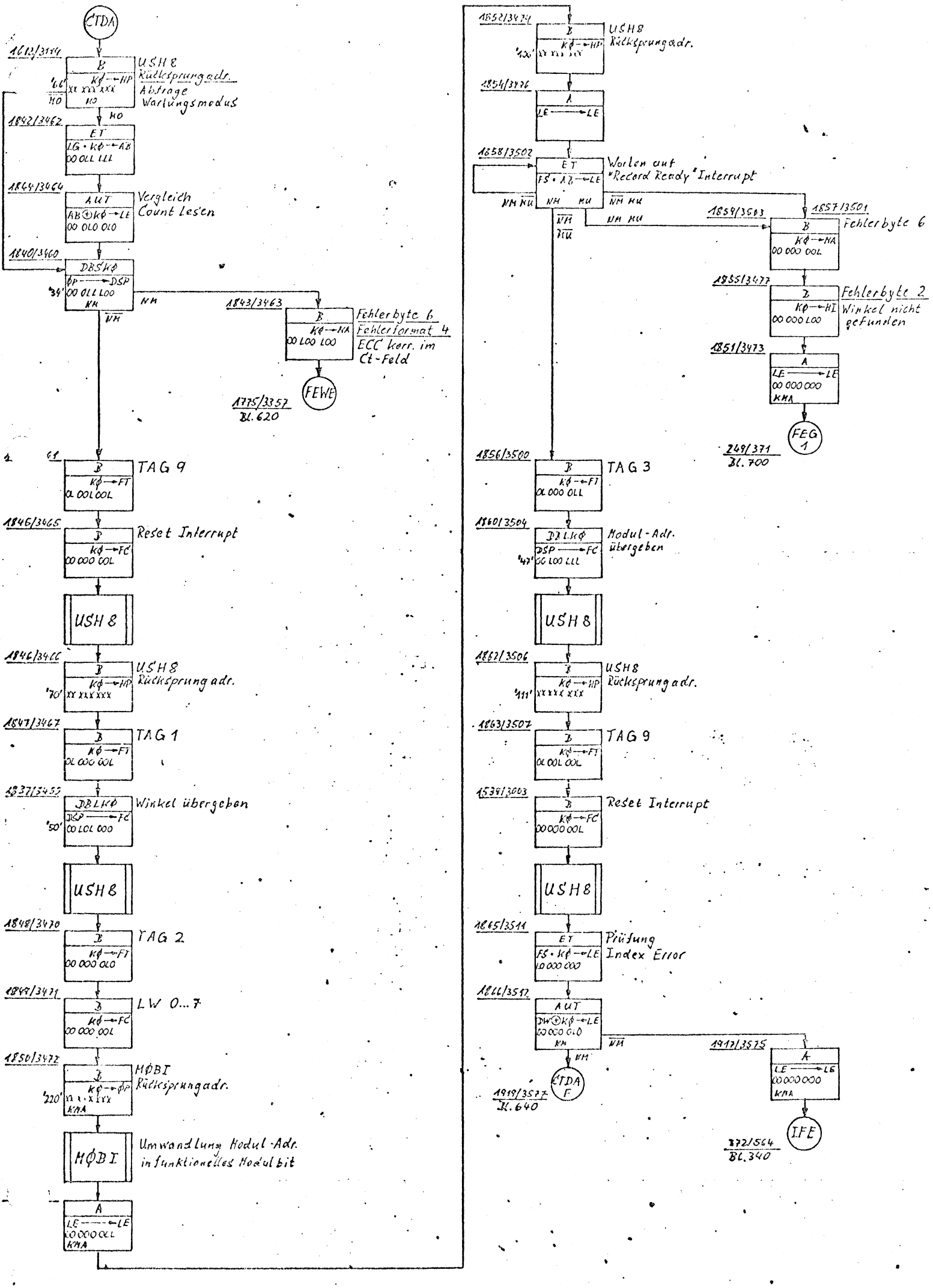

FEØR
1692/3711
A
HE → AB
1708/13254
A
HC → HF
1695/5237
A
PE → HG
1709/3255
B
KØ → HM
XX XXX XXX
'247'
ADDUP Rücksprungadr.
ADDUP
1703/3247
A
HE → ØP
HC
MC
ERKØC
1705/3231
Bl. 580
MC
'250'
B
KØ → HN
00 000 000
ØP7
ØP7
ØP7
1771/3353
ADD
HN + KØ → HN
00 000 L00
1770/3352
A
LE → LE
ØP6
ØP6
ØP6
1731/3303
ADD
HN + KØ → HN
00 000 0L0
1729/3301
A
LE → LE
ØP5
ØP5
ØP5
1707/3253
ADD
HN + KØ → HN
00 000 00L
1706/3252
ADDP
HC + KØ + HC → HC
LL LLL 00L
6 subtrahieren (wegen ECC-Bytes − 7 wegen abgeschnittenen Rest + 1)
Multiplikation mit 2.
Der Fehlerort in Bit (um 4 Bit nach links geschoben Δ · 16 wegen der Wahl der Konstanten) steht in PE, HC, HE. Der Fehlerort in Byte ergibt sich durch Multiplikation mit 2 in PE, HC.
(16·2/256 = 1/8) Der Rest in Bit steht dann in HE (2^7, 2^6, 2^5).
Rest in Bits nach HN
1710/3256
A
LE → LE
HC
HC
HC
1712/3260
ADD
PE + KØ → PE
LL LLL LLL
1711/3257
A
LE → LE
UTH
UTH
ECC-Fehler im ECC-Bytes
1718/3266
UTH
ADD
HC + KØ → HC
00 000 00L
1713/3211
B
KØ → HM
'271'
XXXX XXX
SHPU Rücksprungadr.
SHPU
FEØR
X
ER ECC
1716/3264
B
KØ → HV
00 000 LLL
1657/3163
DBS HN
LE → DSP
NM
NM
1652/3164
NM
ADD
HV + KØ → HN
00 000 00L
1634/3166
AUT
HN ⊕ KØ → LE
G10X XLL
FEØR
X
1721/3271
A
HE → AB
SHPU
1722/3272
A
HE → HF
SHPU
1723/3273
ET
HE • KØ → HG
LL LLL LLO
Herausshiften der 3 Fehlermusterbytes aus H
Da maximal 23 mal ge shiftet werden kann, mus das letzte Bit ausgeble werden.
ERFI
1724/3274
DBS KØ
PE → DSP
00 000 LLL
Fehlerbyte 7
1725/3275
DBS KØ
HC → DSP
00 00L 000
Fehlerbyte 8
1726/3276
DBS KØ
AB → DSP
00 00L 00L
Fehlerbyte 9
1727/3277
DBS KØ
HF → DSP
00 00L 0L0
Fehlerbyte 10
1728/3300
DBS KØ
HG → DSP
00 00L LLL
Fehlerbyte 11
Fehlerort
Fehlermus
FS CØR
1653/3165
DBL KØ
DSP → HE
00 0LL 0LL
1609/3111
A
LE → LE
00 000 L0L
KMA
KØRF
2994/5600
Bl. 610
PE


Korrigierbare Lesefehler
KØRF
2944/5600
B
KØ → HP
'201'
DRAS
Rücksprungadr.
DRAS
2945/5601
ADD
DSP-Anfangsadr.
für Korr. Lesefehler
2946/6602
B
KØ → HP
'203'
ADD2
Rücksprungadr.
ADD2
2947/5603
B
KØ → HP
'213'
HC
ZGELB
Rücksprungadr.
HC
A
LE → LE
ZGELB
2955/5613
B
KØ → HH
HC
2952/5610
A
LE → LE
HHA
1732/3304
DBLKØ
DSP → HC
'32'
1707/3245
DBLKØ
DSP → HØ
'43'
1730/3302
DBLKØ
DSP → HF
'40'
1733/3305
DBLKØ
DSP → HG
'41'
1734/3306
DBLKØ
DSP → HH
'35'
1735/3307
DBLKØ
DSP → HP
'44'
1736/3310
DBLKØ
DSP → PE
'31'
A
DW → LE
2949/5605
B
KØ → HP
'212'
ZGELB
Rücksprungadr.
ZGELB
2954/5612
B
KØ → HH
HC
2950/5606
B
KØ → PE
Fehlerbyte 6
Korr. Fehler-Überlauf
2953/5611
B
KØ → PE
Fehlerbyte 6
Gelesene Bytes-Überlauf
2951/5607
B
KØ → PE
Fehlerbyte 6
Korr. Fehler u
Gelesene Byte
Überlauf
FUEB
2857/5451
Bl. 330
1738/3312
DBLKØ
DSP → CP
'24'
1740/3314
DBLKØ
DSP → HI
'7'
1739/3313
A
HI → LE
KØRFE
1751/3327
Bl. 620
DATA
1741/3315
ET
LG • KØ → DW
1900/3554
AUT
Vergleich
Prüflesen
1901/3555
A
LE → LE
1907/3563
A
LE → LE
Abfrage
Wartungsmodus
1908/3562
D
KØ → HA
Fehlerbyte 6
Fehlerformat 4
ECC korr. im
Datenfeld
FEWE
1775/3357
Bl. 620
1904/3560
A
LE → LE
CYDA
1772/3114
Bl. 630

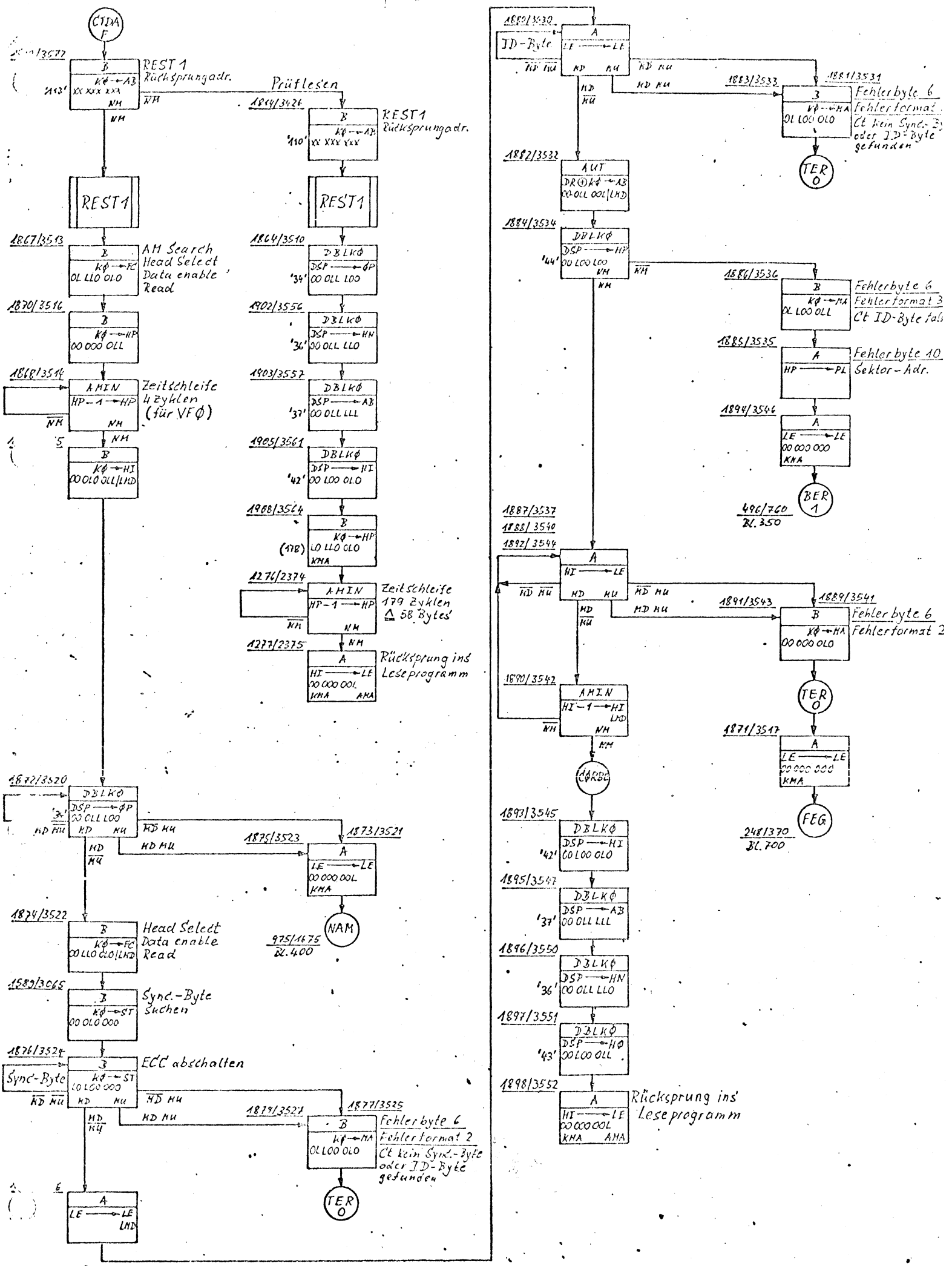
PE


KØRFL
1754/3327
DBLKØ
DSP → HI
10' LO OOL OOO
NM
NH
1747/3323
AMIN
DW − 1 → LE
1745/3321
B
KØ → HN
OO OOL OOL
1743/3317
ADD
HI + KØ → HI
OO OOO LOO
1742/3316
ADD
HI + KØ → LE
LL LLO OLO
1748/3324
ADD
HI + KØ → HI
LL OLL LOO
UTH
1746/3322
Bl. 580
ERNØC
RECØ
1744/3320
DBLHN
DSP → AB
Fehlermusterbytes auslesen
1749/3325
AUT
HN ① KØ → LE
OO OOL LOO
1750/3326
ADD
HN + KØ → HN
OO OOO OOL
1752/3330
AMIN
HI − 1 → HI
RECØ LI
ERECØ
1753/3331
DBSKØ
IIP → DSP
'44' OO LOO LLC
1754/2332
DBSKØ
HØ → DSP
'43' O OOO OLL
1755/3333
ET
LG · KØ → HN
OO OLL LLL
1588/3069
AUT
HN ⊕ KØ → LE
OO OLO OOL
Vergleich RO lesen
1915/3573
AMIN
DW − 1 → LE
HG HN
HG HN
1612/3114
Bl. 630
CTDA
1613/3115
1614/3116
1615/3117
Formatieren
B
KØ → HN
(44) OO LOL LOO
bei HA
1772/3354
ADD
HN + KØ → HN
(104) OL LOL OOO
bei Ct
1773/3355
B
KØ → AB
'30' XX YYY XXX
REST Rücksprungadr.
HO
1776/3360
DBSKØ
ØP → DSP
'34' OO OLL LOO
REST
REØN
1816/3430
AMIN
HN − 1 → HN
Zeitschleife 88 Zyklen bei HA, 208 Zyklen bei Ct. Zum Übertakten von HA bzw HA u. Ct
1818/3432
DBLKØ
DSP → ØP
'34' OO OLL LOO
1817/3431
DBLKØ
DSP → HP
'44' OO LOO LOO
C'ØRBC
1893/3545
Bl. 640
1778/3362
AMIN
DW − 1 → LE
1777/3361
B
KØ → KA
OO OLO LOO
Fehlerbyte 6 ECC korr. im HA-Feld
1774/3356
B
KØ → KA
OO LOO LOO
Fehlerbyte 6 ECC' korr. im Ct-Feld
FEVE
1775/3357
B
KØ → HI
OO 000 000
Fehlerbyte 2
1909/3565
B
KØ → IS
OO OOO OOO
Fehlerbyte 1
1910/3566
B
KØ → PE
OO OCO OOO
Fehlerbyte 0
1911/3567
B
KØ → HC
10 OOC OLO
Fehlereingriffs-zeichen
1899/3553
33LKØ
DSP → HN
'36' OC CLL LLO
3/6 Code
1912/3570
A
LE → LE
O OC OLO
FEA
1024/2000
Bl. 640
Registerkorrekturliste
RECØ LI
Fehlerort kann max. DL+2 sein
0 A LE → LE Spr. → ERECØ 1766/333
1 AUT ØP + AB → PP " ERECØ 1767/333
2 " PE " PE " RECØ 1768/333
3 " FR " FR " " 1769/333
4 " HP " HP " " 1760/333
5 " HM " HM " " 1761/334
6 " HG " HG " " 1762/334
7 " HF " HF " " 1763/334
8 " HS " HS " " 1764/334
9 " HØ " HØ " " 1765/334
10 " HC' " HC " " 1766/334
11 AUT HE ⊕ AB → HE " " 1767/334
12 A LE → LE " " 1768/335
13 A LE → LE Spr. → RECØ 1744/332

PE


CTDA
1842/3462
ET
LG · KØ → AB
00 OLL LLL
1844/3464
AUT
Vergleich
Count Lesen
AB ⊕ KØ → LE
00 OLO OLO
1840/3460
DBS KØ
ØP → DSP
'34' 00 OLL LOO
NN
NN
1843/3463
B
KØ → NA
00 LOO LOO
Fehlerbyte 6
Fehlerformat 4
ECC korr. im Ct-Feld
FEWE
1775/3357
Bl. 620
TAG 9
1846/3465
Reset Interrupt
USH 8
1846/3466
USH 8
Rücksprung adr.
'70' XX XXX XXX
1847/3467
TAG 1
1837/3455
DBL KØ
Winkel übergeben
'50' 00 LOL 000
USH 8
1848/3470
TAG 2
1849/3471
LW 0...7
1850/3472
HØBI
Rücksprungadr.
'220' XX X XXX
HØBI
Umwandlung Modul-Adr. in funktionelles Modulbit
1852/3474
USH 8
Rücksprungadr.
'120' XX XXX XXX
1854/3476
LE → LE
1858/3502
ET
Warten auf "Record Ready" Interrupt
FS · AB → LE
NH NU
NH NU
1859/3503
1857/3501
Fehlerbyte 6
00 000 00L
1855/3477
Fehlerbyte 2
Winkel nicht gefunden
00 000 L00
1851/3473
LE → LE
00 000 000
KMA
FEG 1
249/371
Bl. 700
1856/3500
TAG 3
1860/3504
DBL KØ
Modul-Adr. übergeben
'47' 00 L00 LLL
USH 8
1862/3506
USH 8
Rücksprungadr.
'111' XX XXX XXX
1863/3507
TAG 9
1534/3003
Reset Interrupt
USH 8
1865/3511
ET
Prüfung Index Error
FS · KØ → LE
00 000 000
1866/3512
AUT
DW ⊕ KØ → LE
00 000 0L0
NN
NN
1917/3515
LE → LE
00 000 000
KMA
CTDA F
1919/3517
Bl. 640
IFE
372/564
Bl. 340

PE


CTDA F
B
KØ → AB
XX XXX XXX
NM
REST 1
Rücksprungadr.
Prüflesen
REST1
AM Search
Head Select
Data enable
Read
DBLKØ
DSP → ØP
00 OLL LOO
DSP → HP
00 000 OLL
AMIN
HP − 1 → HP
Zeitschleife
4 Zyklen
(für VFØ)
DSP → HI
00 OLO OLL/LHD
DSP → HN
00 LOO LOO
DSP → AB
00 OLL LLL
DSP → HI
00 LOO OLO
(118)
KMA
Zeitschleife
179 Zyklen
≙ 58 Bytes
A
HI → LE
00 000 00L
AMA
Rücksprung ins
Leseprogramm
1872/3520
1873/3521
1874/3522
1875/3523
1876/3524
1877/3525
1879/3527
1880/3530
1881/3531
1882/3532
1883/3533
1884/3534
1885/3535
1886/3536
1887/3537
1888/3540
1889/3541
1890/3542
1891/3543
1892/3544
1893/3545
1894/3546
1895/3547
1896/3550
1897/3551
1898/3552
ID-Byte
LE → LE
HD HU
AUT
DR ⊕ KØ → AB
00 OLL 00L/LHD
3
KØ → MA
OL LOO OLO
Fehlerbyte 6
Fehlerformat 3
Ct kein Sync.-Byte
oder ID-Byte
gefunden
Fehlerbyte 10
Sektor-Adr.
00 000 000
KMA
Fehlerbyte 6
Fehlerformat 2
00 000 OLO
CØRDL
TER 0
FEG
496/760
Bl. 350
975/1675
Bl. 400
248/370
Bl. 700
NAM
Head Select
Data enable
Read
00 LLO OLO/LHD
Sync.-Byte
Suchen
00 OLO 000
ECC abschalten
Sync-Byte
LO LOO 000
LE → LE
LHD

Allgemeine Fehlerabhandlung

FEB

Versorgung: PE : Fehlerbyte 0
IS : " 1
HI : " 2
MA : " 6
HD : Fehlerbyte 16 (Sektoranzahl falls relevant)

1024/2000 DBSKØ PE → DSP 00 000 000 — Fehlerbyte 0
1025/2001 DBSKØ IS → DSP 00 000 001 — Fehlerbyte 1
1026/2002 DBSKØ MA → DSP 00 000 110 — Fehlerbyte 6
1027/2003 ET MA + FS → MA 00 001 111
1028/2004 BUT MA ⊕ Ø → LE 00 000 150 — Vergleich Fehlerformat 4
1031/2007 DBSKØ MI → DSP 00 000 010 — Fehlerbyte 2
1032/2010 DBSKØ FT → DSP 00 001 000 — Fehlerbyte 8
1030/2006 DBSKØ FS → DSP 00 001 010 — Fehlerbyte 10
1140/2164 ET FT + KØ → FT 00 111 111
1141/2165 ADD FT + KØ → FT 01 000 000
1142/2166 DBSKØ FC → DSP 00 001 001 — Fehlerbyte 9
1143/2167 B IC → LE
1127/2147 DBSKØ FS → DSP 00 000 111 — Fehlerbyte 7
1033/2011 BUT FT ⊕ Ø → LE 01 001 011 — Vergleich TAG 11
1037/2046 BUT FT ⊕ KØ → LE 01 000 101 — Vergleich TAG 5
1055/2037 B KØ → FC 00 010 110
1054/2036 B KØ → HP XX XXX XXX — UPTFE Rücksprungadr. '42'
1056/2040 B KØ → FT 00 000 101 — TAG 5

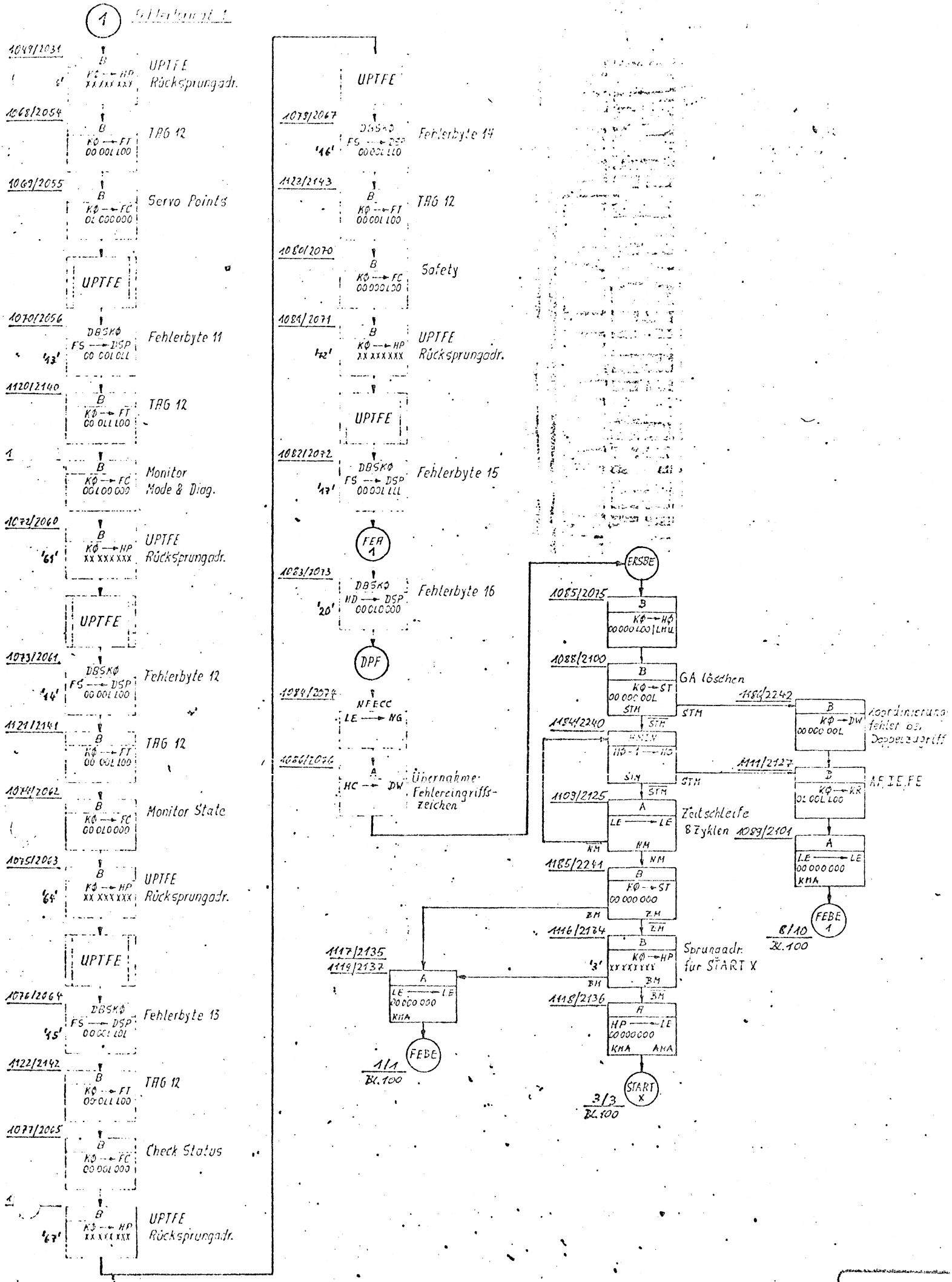
1046/2026 DBSKØ LE ← DSP 00 000 010 '252'

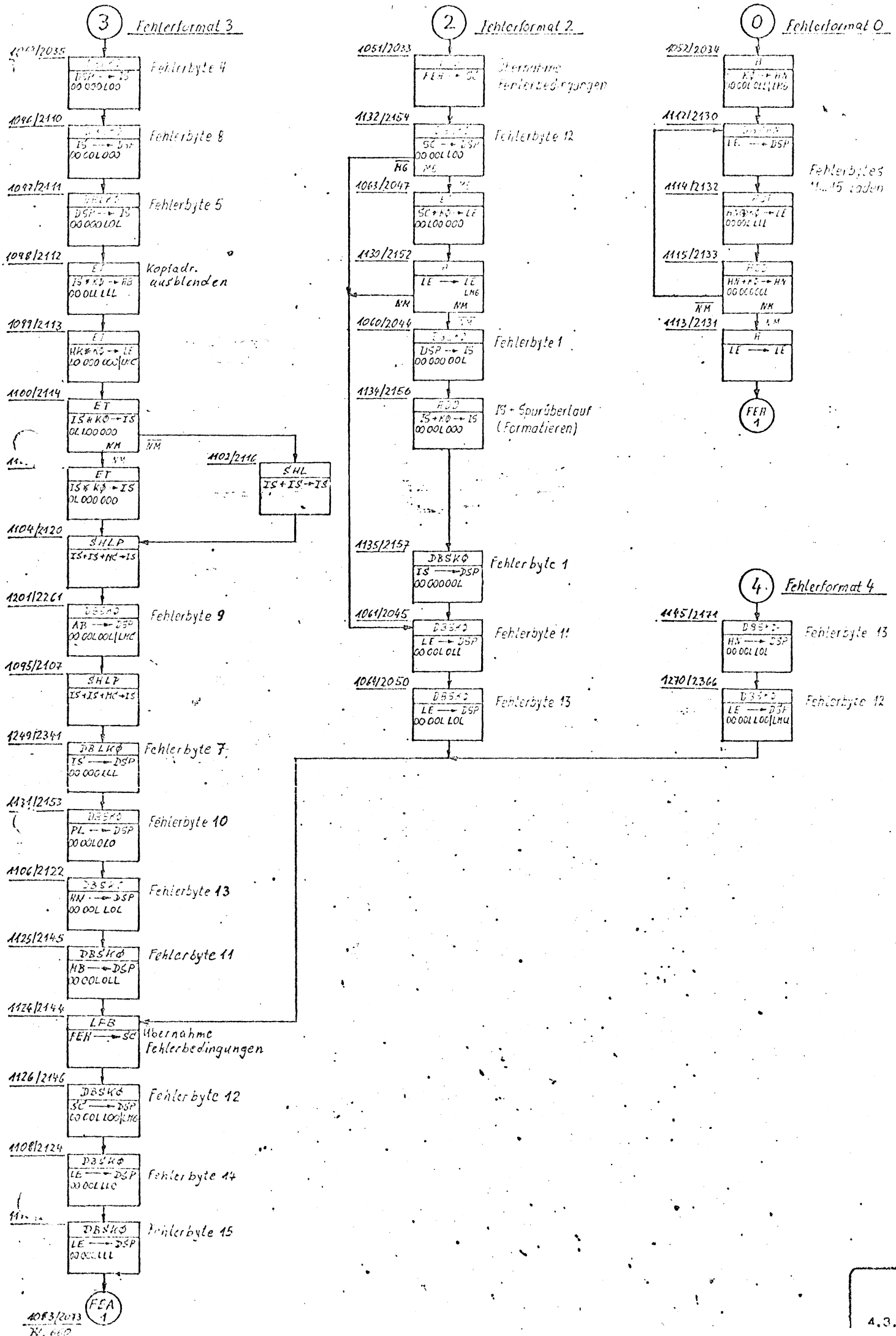
1044/2024 B KØ → HP XX XXX XXX — UPTFE Rücksprungadr. '40'

1045/2021 B KØ → FC 00 001 000 — Read CAR
UPTFE
1034/2012 DBSKØ FS → DSP 00 000 100 — Fehlerbyte 4
1059/2043 BUT FT ⊕ KØ → FT 00 010 000 — TAG 5
1035/2013 B KØ → FC 00 001 000 — Read MAR
UPTFE
1040/2020 DBSKØ FS → DSP 00 000 101 — Fehlerbyte 5
1058/2042 B KØ → HP XX XXX XXX '53' — UPTFE Rücksprungadr.
1066/2062 B KØ → FT 00 001 001 — TAG 9
1065/2051 B KØ → FC 00 000 100 — Reset Read
UPTFE
1067/2053 B KØ → FT 10 001 001 — TAG 9
1042/2022 B KØ → ST 00 000 000
1043/2023 BUT MA ⊕ Ø → LE 00 000 100 — Vergleich Fehlerformat 4
1057/2041 B KØ → FT 00 010 001 — TAG 5

1045/2025 DBSKØ FS → DSP 00 000 011 — Fehlerbyte 3
NM NM
1144/2170 B KØ → FC 01 001 001 — TAG 9 1145/2171 BL. 670 — (4) Fehlerformat 4
1146/2172 DBI KØ DSP → OP 00 000 111 — Fehlerbyte 7
1147/2173 ET ØP + KØ → LE 00 100 000 — Prüfung Device Check (Bit 5)
1148/2174 ET FS + KØ → AB 00 100 000
TM NM
1151/2177 ADD ØP + AB → ØP
1149/2175 DBSKØ ØP → DSP 00 000 111 — Fehlerbyte 7
1150/2176 BUT MA ⊕ Ø → LE 00 000 001 — Vergleich Fehlerformat 1
1047/2027 BUT MA ⊕ KØ → LE 00 000 010 — Vergleich Fehlerformat 2
NM NM NM — (1) Fehlerformat 1 — 1049/2031 BL. 660
1048/2030 BUT MA ⊕ Ø → LE 00 000 011 — Vergleich Fehlerformat 3
NM NM NM — (2) Fehlerformat 2 — 1051/2033 BL. 670
1050/2032 B MA → LE
NM NM NM — (3) Fehlerformat 3 — 1053/2035 BL. 670
(0) Fehlerformat 0 — 1052/2034 BL. 670

FEB

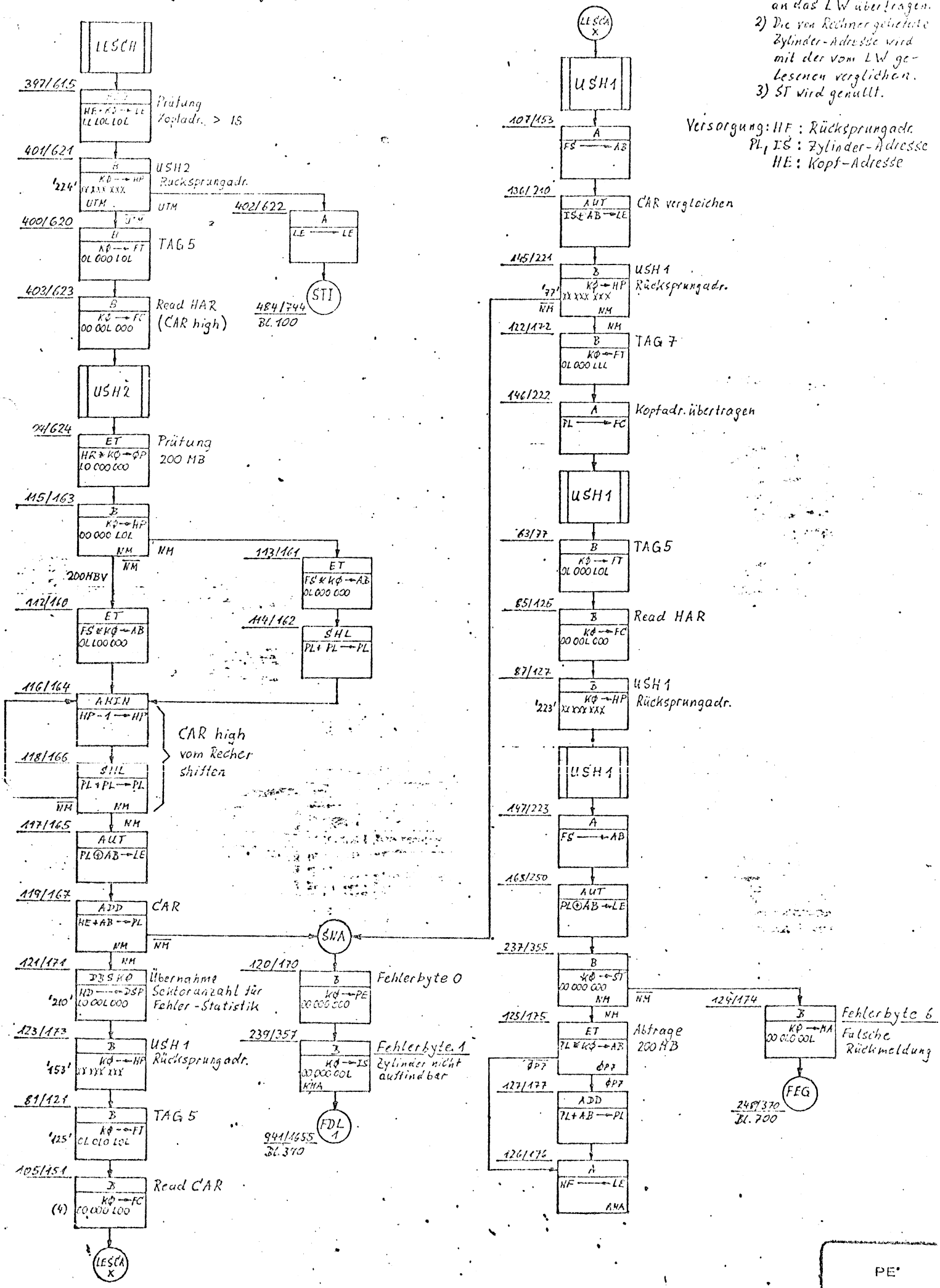

1
Fortsetzung 1
1069/2034
UPTFE
Rücksprungadr.
1068/2054
TAG 12
1069/2055
Servo Points
UPTFE
1070/2056
Fehlerbyte 11
'13'
1120/2140
TAG 12
1
Monitor
Mode & Diag.
1072/2060
'61'
UPTFE
Rücksprungadr.
UPTFE
1073/2061
Fehlerbyte 12
'14'
1121/2141
TAG 12
1074/2062
Monitor State
1075/2063
'65'
UPTFE
Rücksprungadr.
UPTFE
1076/2064
Fehlerbyte 13
'15'
1122/2142
TAG 12
1077/2065
Check Status
'67'
UPTFE
Rücksprungadr.
UPTFE
1079/2067
Fehlerbyte 14
'16'
1123/2143
TAG 12
1080/2070
Safety
1081/2071
'12'
UPTFE
Rücksprungadr.
UPTFE
1082/2072
Fehlerbyte 15
'17'
FEA 1
1083/2073
Fehlerbyte 16
'20'
DPF
1084/2074
1086/2076
Übernahme Fehlereingriffs-zeichen
ERSBE
1085/2075
1088/2100
GA löschen
1184/2242
Koordinierungs-fehler bei Doppelzugriff
1154/2240
1111/2127
AF IE FE
1103/2125
Zeitschleife
8 Zyklen
1089/2101
1185/2241
1116/2134
Sprungadr. für START X
1117/2135
1119/2137
1118/2136
FEBE
1/1
Bl. 100
START X
3/3
Bl. 100
FEBE 1
8/10
Bl. 100


3 Fehlerformat 3
1095/2035
Fehlerbyte 4
1096/2110
Fehlerbyte 8
1097/2111
Fehlerbyte 5
1098/2112
Kopfadr. ausblenden
1099/2113
1100/2114
1103/2116
SHL
IS+IS→IS
1104/2120
SHLP
1201/2261
Fehlerbyte 9
1095/2107
SHLP
1249/2341
Fehlerbyte 7
1131/2153
Fehlerbyte 10
1106/2122
Fehlerbyte 13
1125/2145
Fehlerbyte 11
1124/2144
LFB
FEH → SC
Übernahme Fehlerbedingungen
1126/2146
Fehlerbyte 12
1108/2124
Fehlerbyte 14
Fehlerbyte 15
FEA 1
1083/2013
2 Fehlerformat 2
1051/2033
Übernahme Fehlerbedingungen
1132/2154
Fehlerbyte 12
1063/2047
1130/2152
1060/2044
Fehlerbyte 1
1134/2156
IS + Spurüberlauf (Formatieren)
1135/2157
Fehlerbyte 1
1061/2045
Fehlerbyte 11
1064/2050
Fehlerbyte 13
0 Fehlerformat 0
1052/2034
1112/2130
1114/2132
Fehlerbytes 11...15 laden
1115/2133
1113/2131
FEA 1
4 Fehlerformat 4
1145/2171
Fehlerbyte 13
1270/2366
Fehlerbyte 12

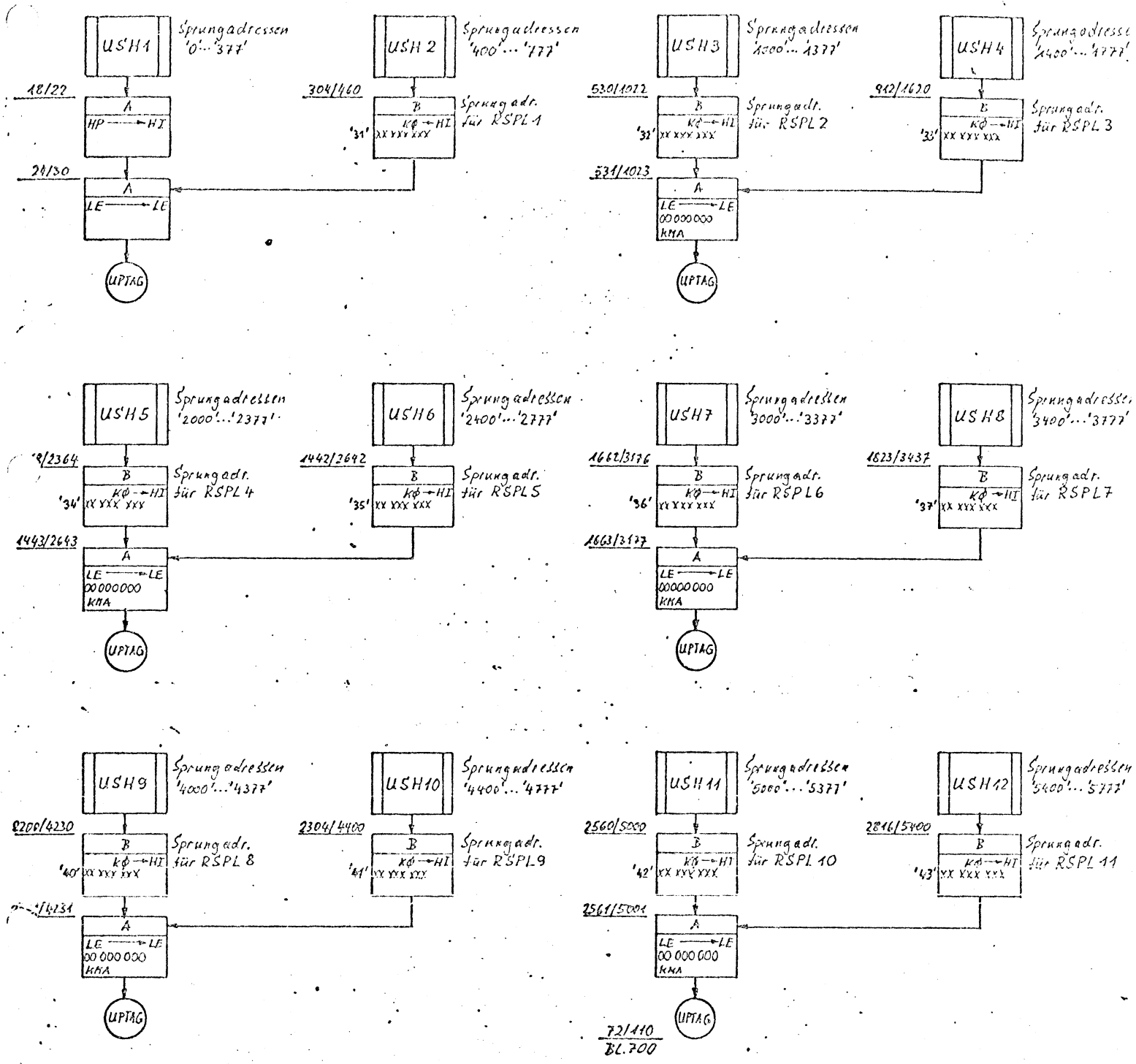
Überprüfung auf richtige Zylinder und Kopfadresse

Leistung: 1) Die Kopf-Adresse wird auf ≤ IS überprüft und an das LW übertragen.
2) Die von Rechner gelieferte Zylinder-Adresse wird mit der vom LW gelesenen verglichen.
3) ST wird genullt.

Versorgung: HF : Rücksprungadr.
PL, IS : Zylinder-Adresse
HE : Kopf-Adresse

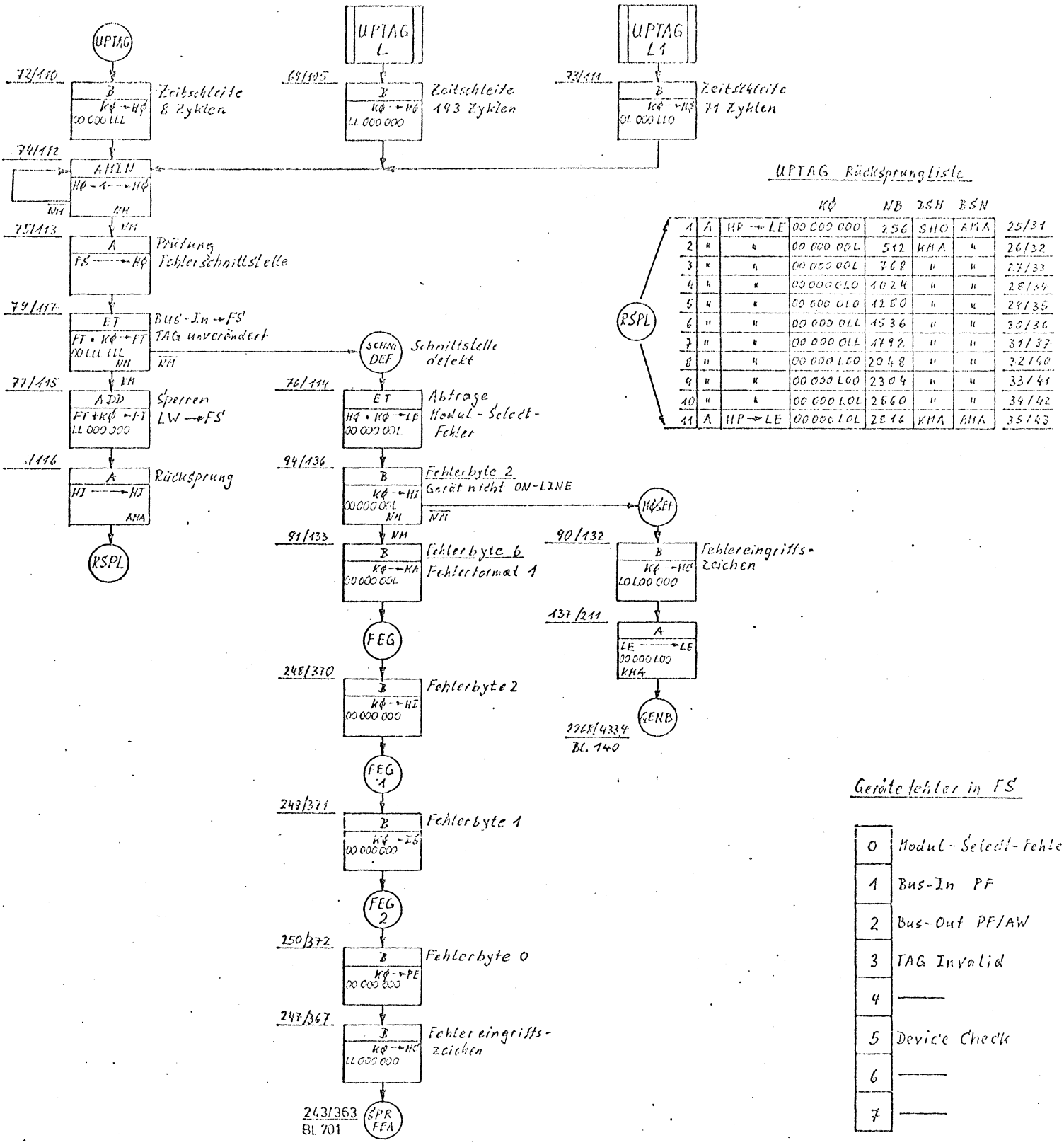


Unterprogramm zum Warten auf Rückantwort nach einem TAG und Fehlerabfrage



Leistung: Es wird 4 µs gewartet (bei UPTAGL ca. 93 µs, bei UPTAGL1 ca. 29 µs), danach die Fehlerschnittstelle abgefragt und der Bus-In durchgeschaltet.

Versorgung: FT 7,6 = 0L
HP: Rücksprungadr.
Bei UPTAGL bzw. UPTAGL1 muss noch HI entsprechend der Rücksprungseite (siehe Liste RSPL Bl. 620) geladen werden.

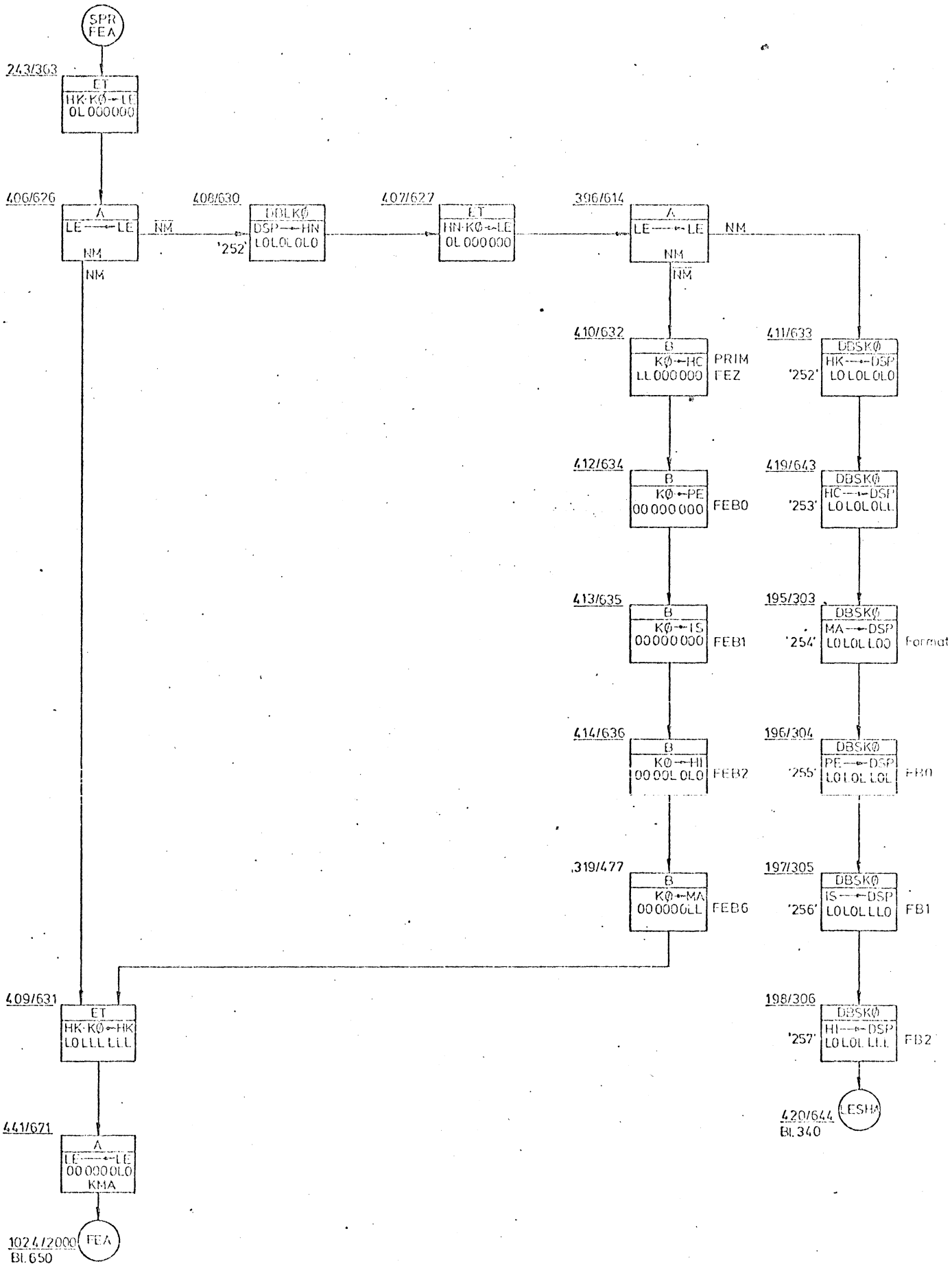


UPTAG Rücksprungliste

			Kϕ	NB	BSH	BSN	
1	A	HP → LE	00 000 000	256	SHO	AHA	25/31
2	"	"	00 000 00L	512	KHA	"	26/32
3	"	"	00 000 0LL	768	"	"	27/33
4	"	"	00 000 LLO	1024	"	"	28/34
5	"	"	00 000 LLO	1280	"	"	29/35
6	"	"	00 000 LLL	1536	"	"	30/36
7	"	"	00 000 LLL	1792	"	"	31/37
8	"	"	00 000 L00	2048	"	"	32/40
9	"	"	00 000 L00	2304	"	"	33/41
10	"	"	00 000 L0L	2560	"	"	34/42
11	A	HP → LE	00 000 L0L	2816	KHA	AHA	35/43

Geräte Fehler in FS

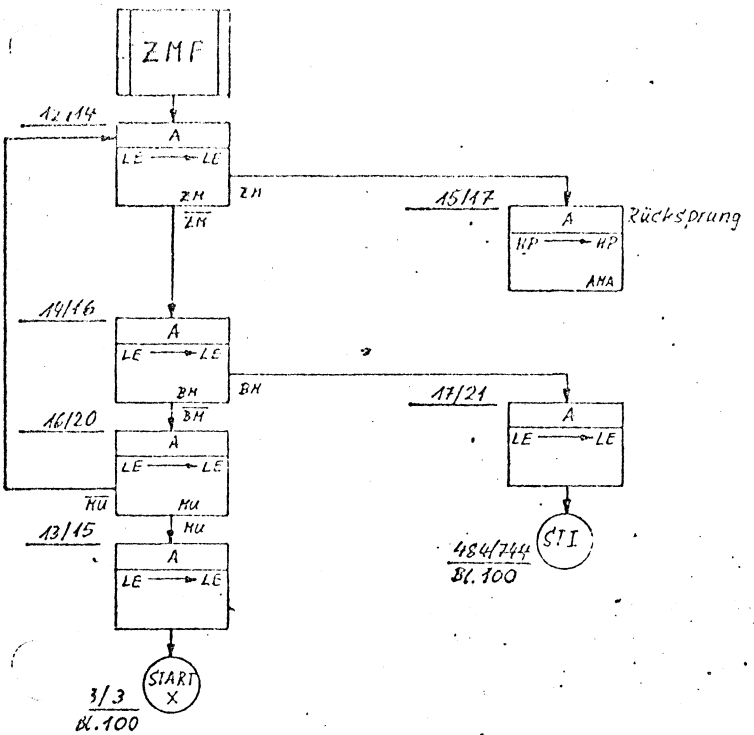
0	Modul-Select-Fehler
1	Bus-In PF
2	Bus-Out PF/AW
3	TAG Invalid
4	——
5	Device Check
6	——
7	——



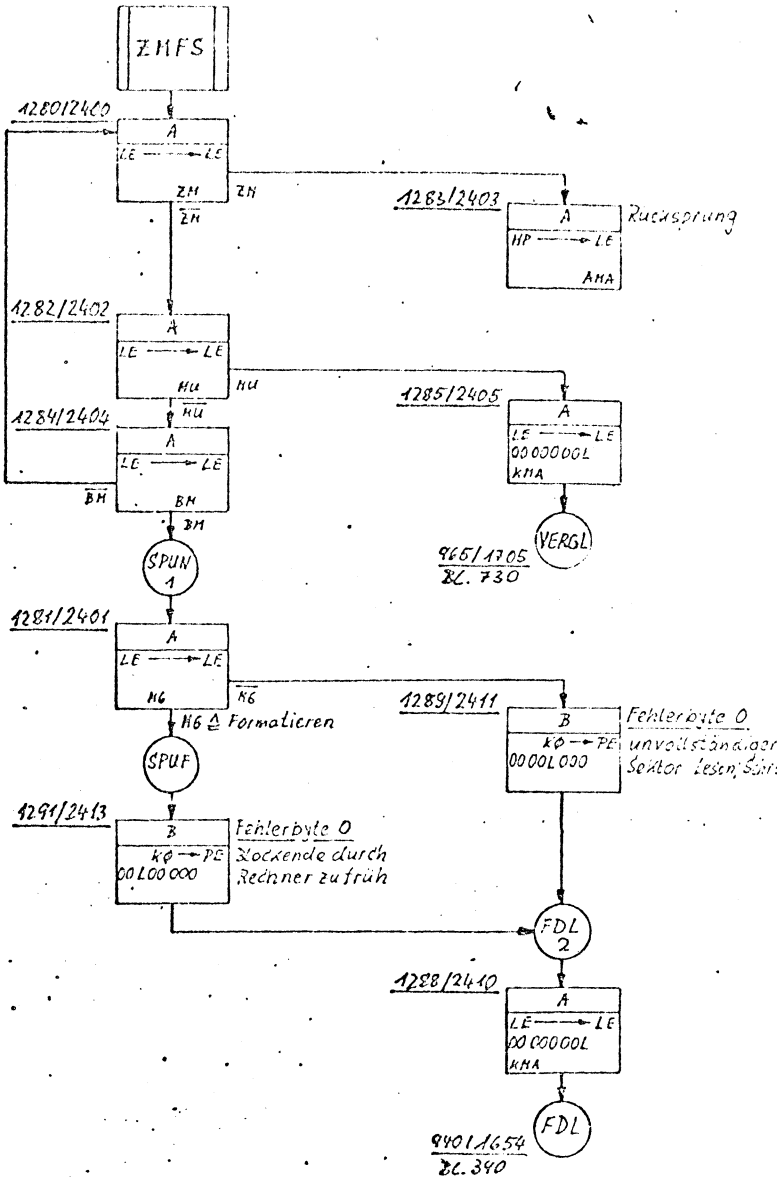
ae-Stand: "A"-0477

PE

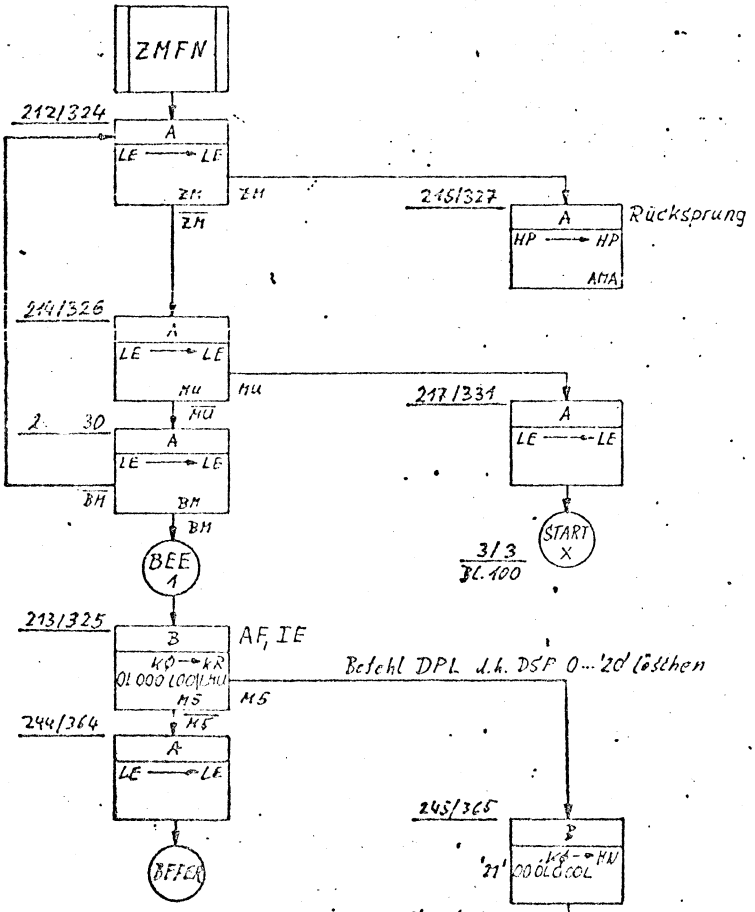
Warten auf ZM



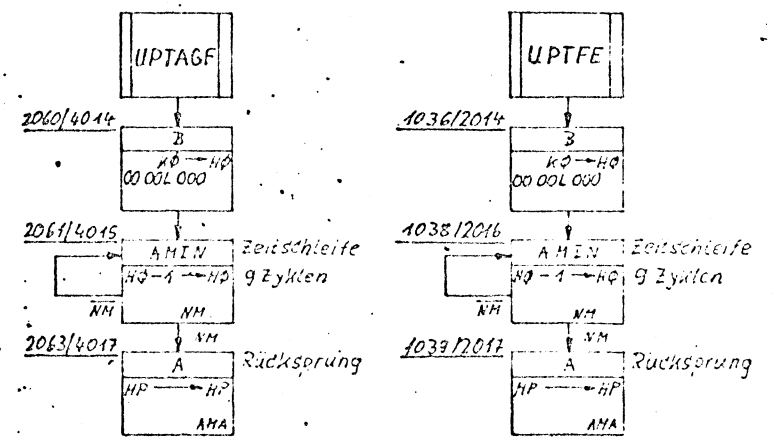
Warten auf ZM bei "Schreiben"



Warten auf BM



Warten nach TAG ohne Fehlerabfrage



Ermitteln 3/6 Code Adresse

DRAS

Leistung: 1) 3/6 Code-Adr. wird nach DSP-Adr. 3 geschrieben
2) In HI steht nach Durchlauf die DSP-Anfangsadr. für Statistikdaten des angesprochenen Laufwerks.

Versorgung: HP: Rücksprungadresse

1/5404 ET FT • KØ → FT 00 LLL LLL

2821/5405 AUT FT ⊕ KØ → FT LO 000 000

2822/5406 B KØ → HI 00 LOL LOL '55' DSP-Anfangsadr. Statistikbytes → HI

2823/5407 DBSKØ FS → DSP 00 000 OLL

2824/5410 ET FS • KØ → ØP 00 000 LLL

2825/5411 ET FT • KØ → FT 00 LLL LLL

20. 12 AMIN ØP − 1 → ØP NM NN

2828/5414 NM ADD HI + KØ → HI 00 00L OLL DSP-Adr. entsprechend Laufwerk hochzählen

2827/5413 A ØP → LE

2829/5415 A HP → LE AMA Rücksprung

Addieren über 2 Register

ADD2

Leistung: Ein Zähler bestehend aus 2 DSP-Zellen (Adresse in HN) wird um 1 erhöht. Läuft der Zähler über, ist beim Rücksprung HC gesetzt.

Versorgung: HP: Rücksprungadresse
HN: DSP-Adr. des niederwertigen Bytes des Zählers.

2832/5420 DBLHN DSP → AB

2833/5421 ADD AB + KØ → AB 00 000 00L

2834/5422 DBSHN AB → DSP LHC UTM UTM

2838/5426 UTM AMIN HN − 1 → HN

2836/5424 A HP → LE AMA Rücksprung

2840/5430 DBLHN DSP → AB

2841/5431 ADD AB + KØ → AB 00 000 00L

2835/5423 DBSHN AB → DSP UTM UTM

2837/5425 UTM A HP → LE AMA Rücksprung

2839/5427 A HP → LE SHC AMA Rücksprung

Durchschalten der Geräteinterrupts

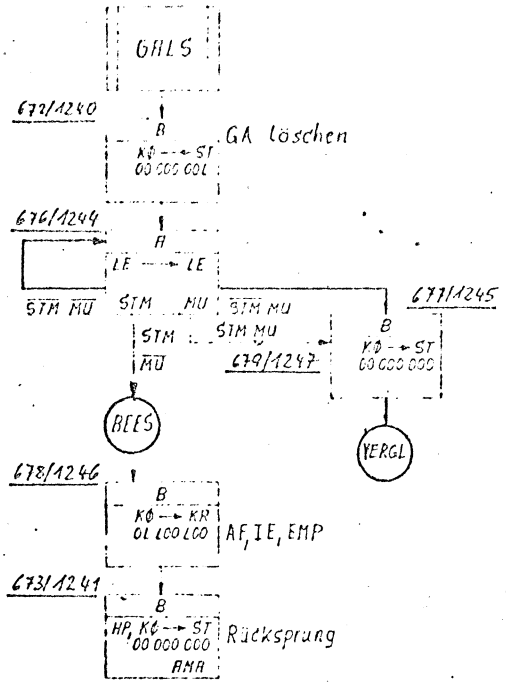
TAG2

2180/4204 B KØ → FT 0L CL0 0L0 TAG 2 Fehler durchschalten

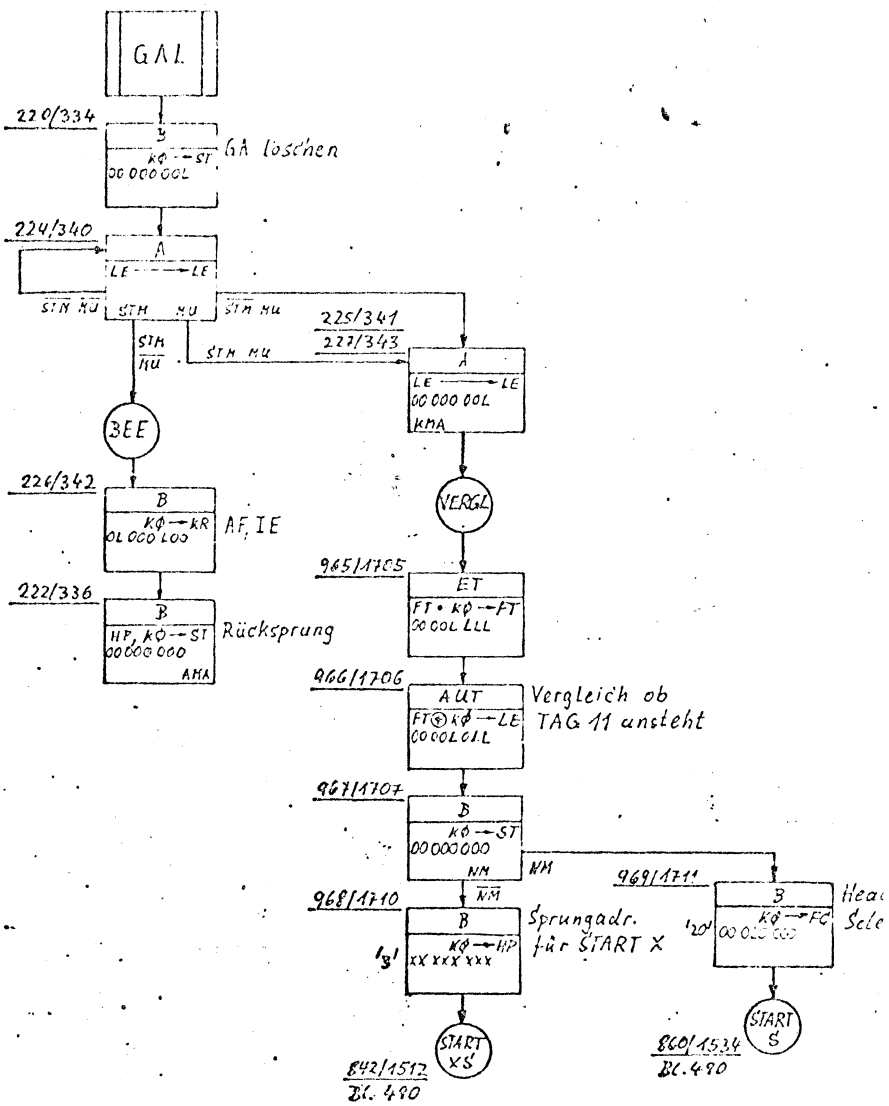
2181/4205 B KØ → FC 00 000 00L LW 0…7

USH9

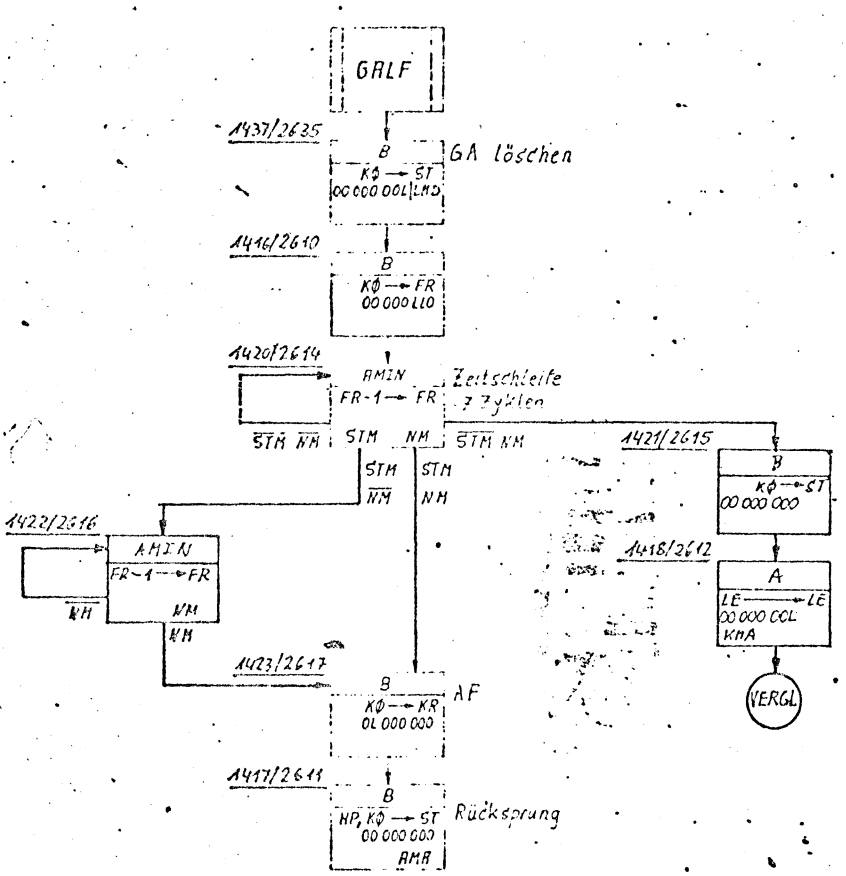
GA-Schleife löschen bei "Schreiben"



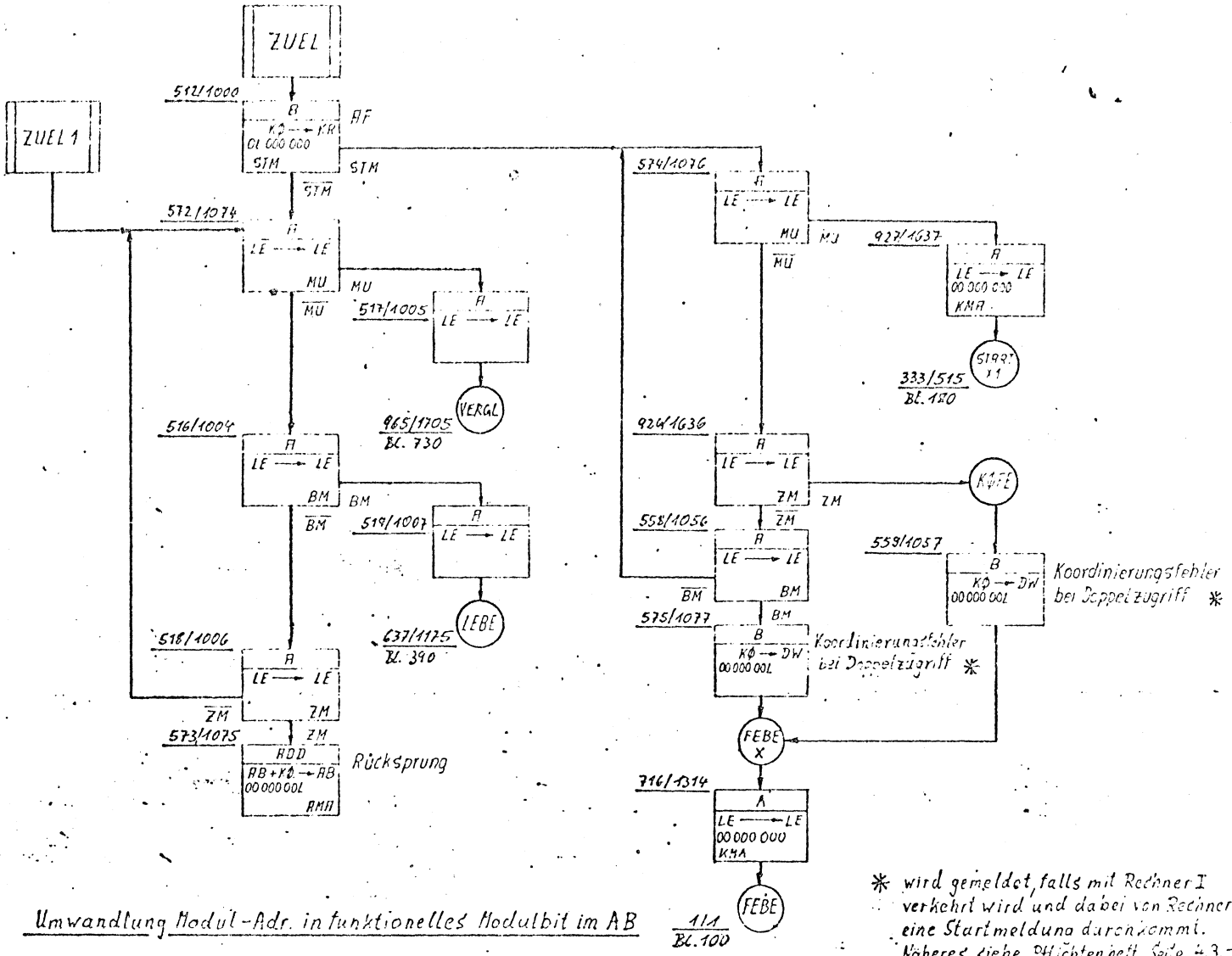
GA-Schleife löschen



GA-Schleife löschen bei "Formatieren"

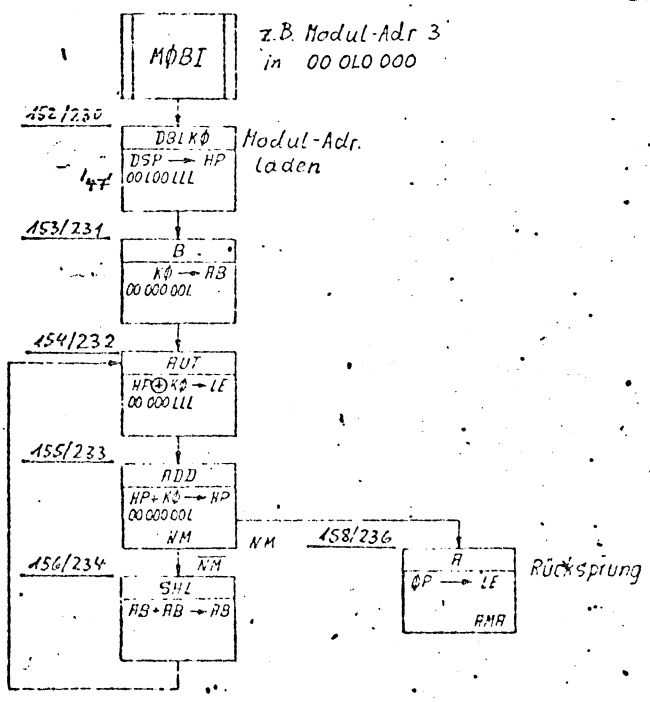


ZM-Abfrage und Quittierung beim Übertragen von Daten (IIA und CI) zum Rechner

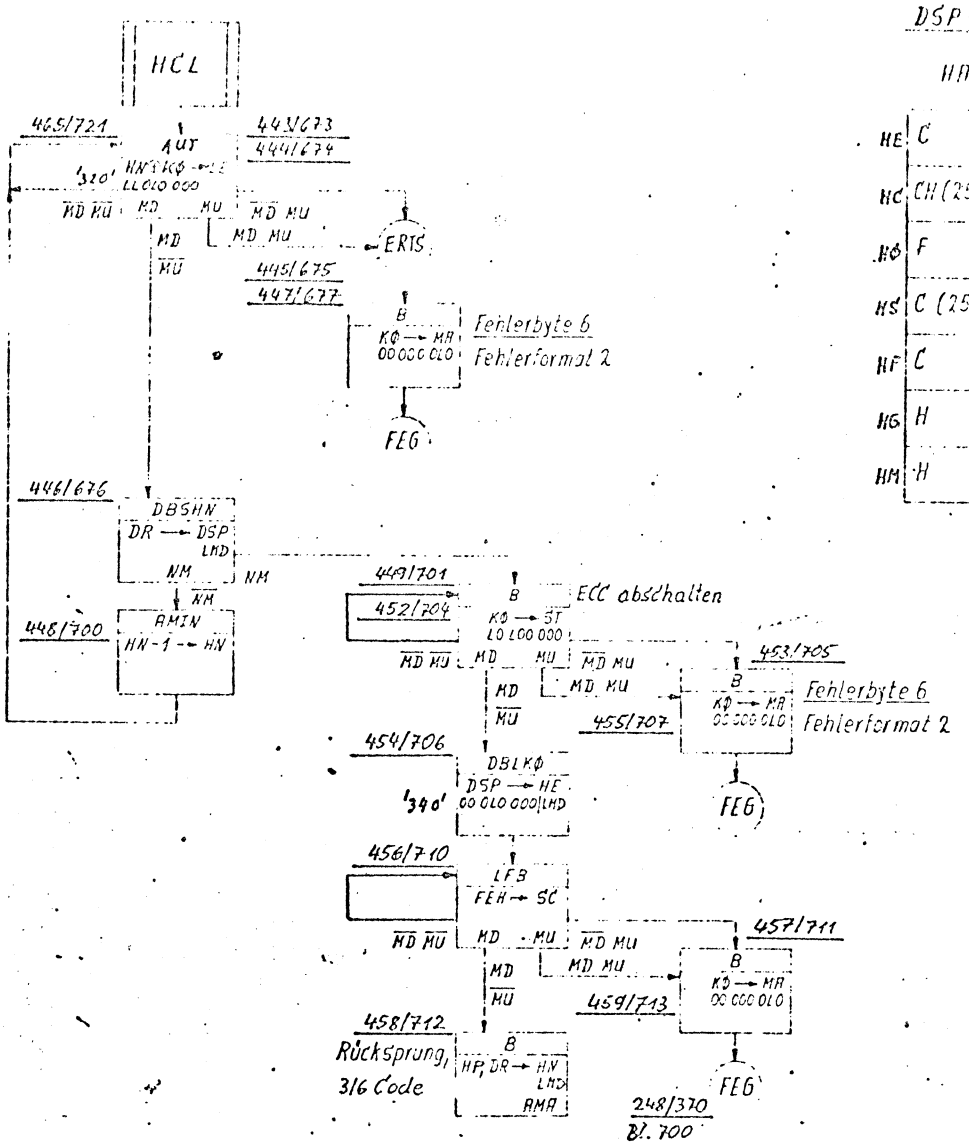


* wird gemeldet, falls mit Rechner I verkehrt wird und dabei von Rechner II eine Startmeldung durchkommt. Näheres siehe Pflichtenheft Seite 4.3.-450

Umwandlung Modul-Adr. in funktionelles Modulbit im AB



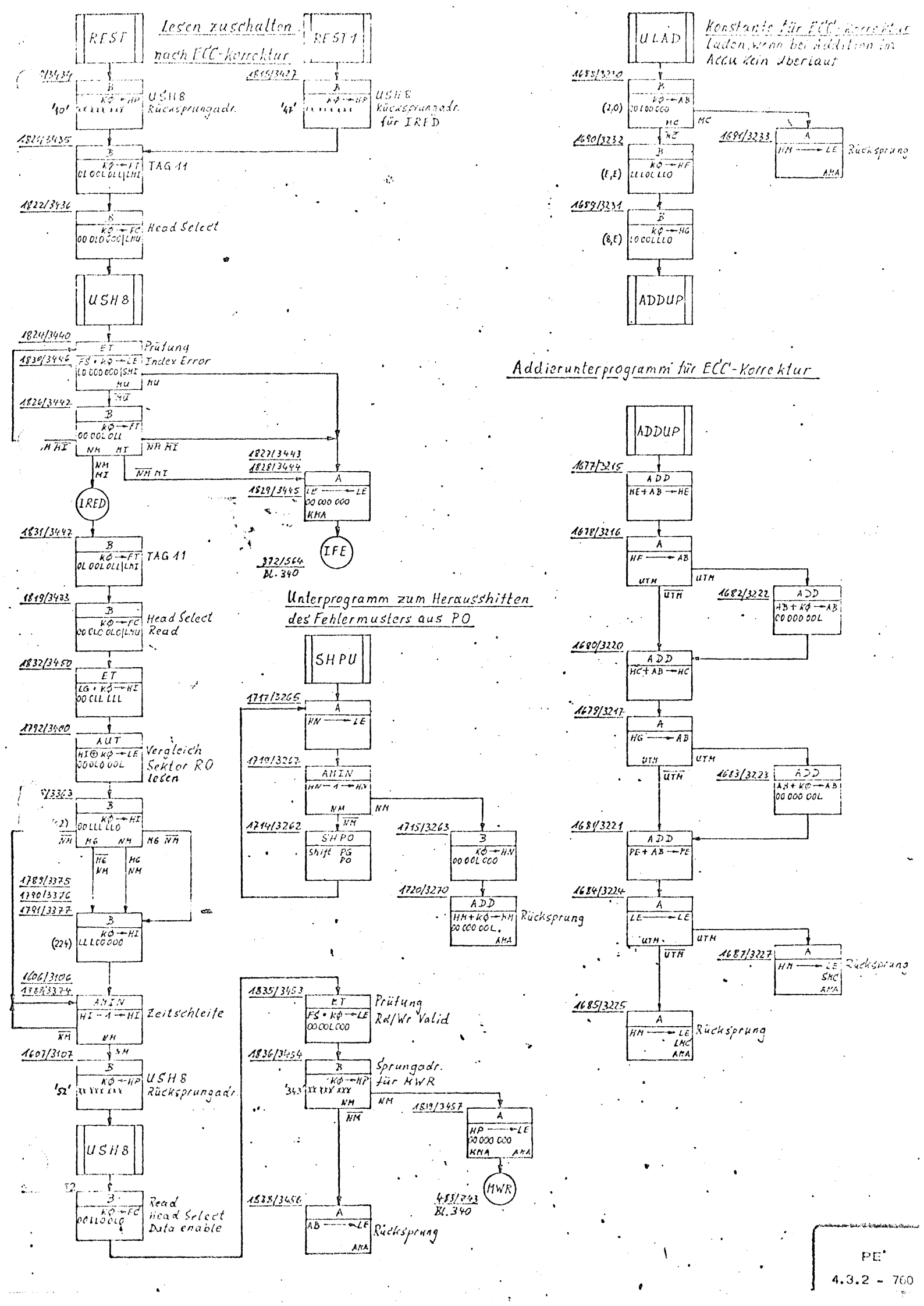
HA- und Ct-Feld lesen

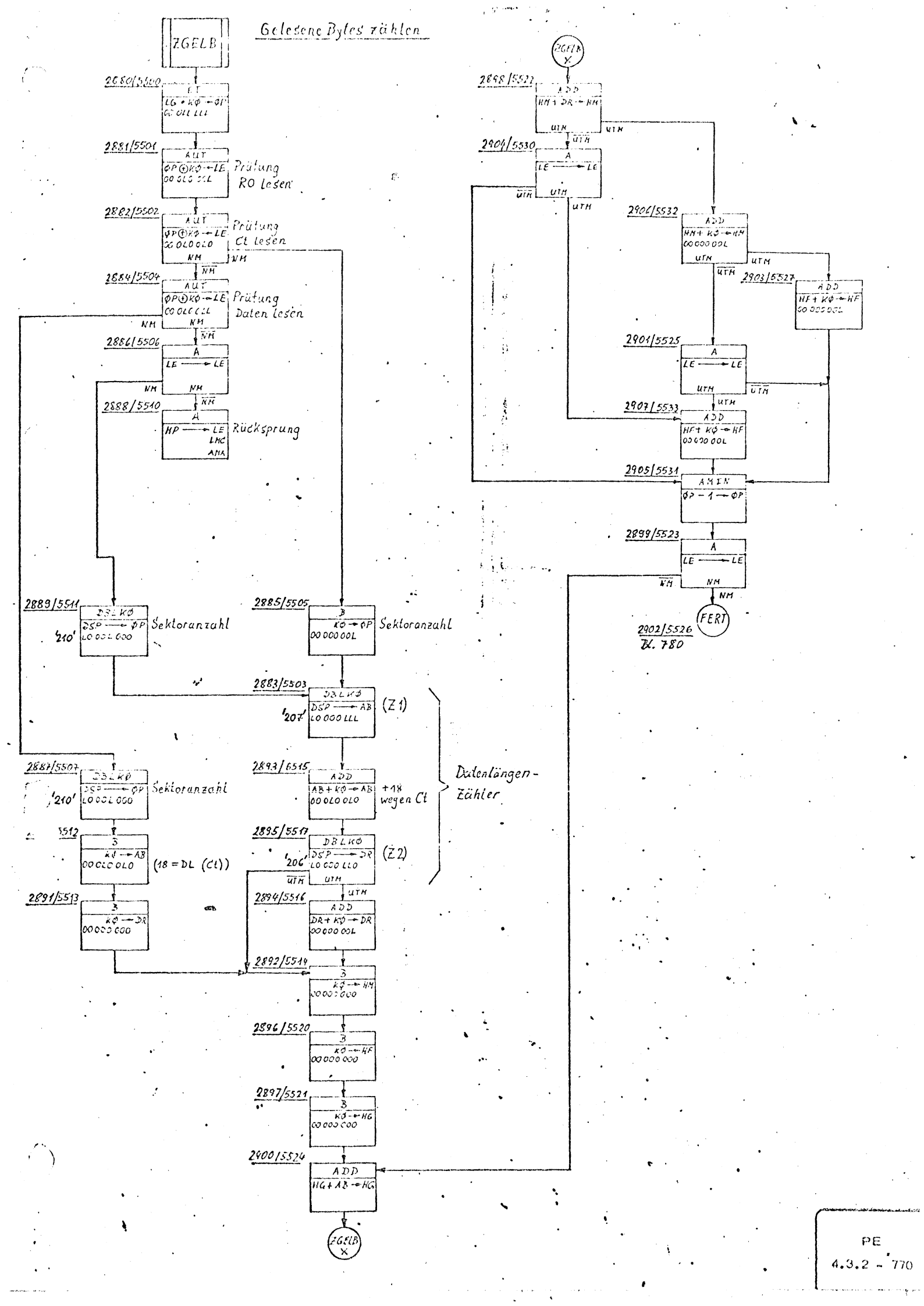


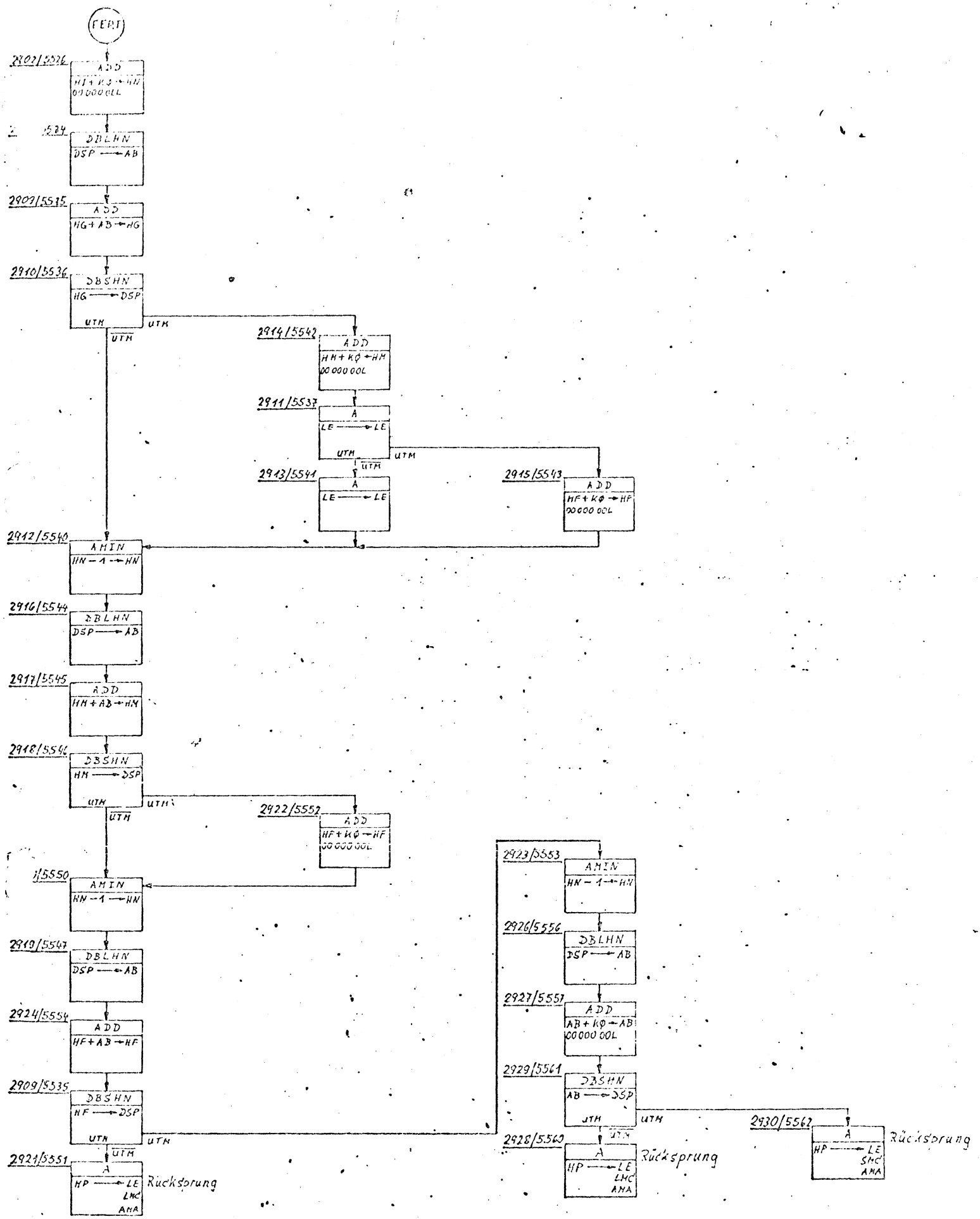
DSP - Belegung

HA			CT		
HE	C	'334'	HE	C	'340'
HC	CH (256)	'333'	HC	CH (256)	'337'
HØ	F	'332'	HØ	F	'336'
HS	C (256)	'331'	HS	C (256)	'335'
HF	C	'330'	HF	C	'334'
HG	H	'327'	HG	H (256)	'333'
HM	H	'326'	HM	H	'332'
			HP	R	'331'
			FR	KL	'330'
			PE	D	'327'
			ØP	L	'326'

HA bzw. Ct inclusive ECC-Bytes wird gelesen und entsprechend der obigen Belegung im Datenspeicher abgelegt.


REST
Lesen zuschalten nach ECC-korrektur
REST1
ULAD
Konstante für ECC-korrektur laden, wenn bei Addition im Accu kein Überlauf
USH8
Rücksprungadr.
USH8
Rücksprungadr. für IRED
TAG 11
Head Select
Prüfung
Index Error
IRED
IFE
Unterprogramm zum Herausshiften des Fehlermusters aus PO
SHPU
SHPO
Shift PG PO
Vergleich Sektor RO lesen
Zeitschleife
Read Head Select Data enable
Prüfung Rd/Wr Valid
Sprungadr. für MWR
MWR
Rücksprung
ADDUP
Addierunterprogramm für ECC-Korrektur
ADD
AMIN
PE
4.3.2 - 760


Gelesene Bytes zählen
ZGELB
2880/5500
LT
LG • KØ → ØP
2881/5501
AUT
ØP ⊕ KØ → LE
Prüfung RO lesen
2882/5502
AUT
Prüfung Ct lesen
2884/5504
AUT
Prüfung Daten lesen
2886/5506
A
LE → LE
2888/5510
A
HP → LE
Rücksprung
2889/5511
DBL KØ
DSP → ØP
'210'
Sektoranzahl
2885/5505
B
KØ → ØP
Sektoranzahl
2883/5503
DBL KØ
DSP → AB
'207'
(Z1)
2887/5507
DBL KØ
DSP → ØP
'210'
Sektoranzahl
2893/6515
ADD
AB + KØ → AB
+18 wegen Ct
Datenlängen-Zähler
5512
B
KØ → AB
(18 = DL (Ct))
2895/5517
DBL KØ
DSP → DR
'206'
(Z2)
2891/5513
B
KØ → DR
2894/5516
ADD
DR + KØ → DR
2892/5514
B
KØ → NM
2896/5520
B
KØ → HF
2897/5521
B
KØ → HG
2900/5524
ADD
HG + AB → HG
ZGELB X
2898/5522
ADD
HM + DR → HM
2904/5530
A
LE → LE
2906/5532
ADD
HM + KØ → HM
2903/5527
ADD
HF + KØ → HF
2901/5525
A
LE → LE
2907/5533
ADD
HF + KØ → HF
2905/5531
AMIN
ØP − 1 → ØP
2899/5523
A
LE → LE
FERT
2902/5526
K. 780
PE
4.3.2 - 770


FERT
2902/5536
ADD
HH + KØ → HH
00 000 00L
5534
DBLHN
DSP → AB
2903/5535
ADD
HG + AB → HG
2910/5536
DBSHN
HG → DSP
UTM
UTM
UTM
2914/5542
ADD
HH + KØ → HH
00 000 00L
2911/5537
A
LE → LE
UTM
UTM
UTM
2913/5541
A
LE → LE
2915/5543
ADD
HF + KØ → HF
00 000 00L
2912/5540
AMIN
HN − 1 → HN
2916/5544
DBLHN
DSP → AB
2917/5545
ADD
HH + AB → HH
2918/5546
DBSHN
HH → DSP
UTM
UTM
UTM
2922/5552
ADD
HF + KØ → HF
00 000 00L
5550
AMIN
HN − 1 → HN
2919/5547
DBLHN
DSP → AB
2924/5554
ADD
HF + AB → HF
2909/5535
DBSHN
HF → DSP
UTM
UTM
UTM
2921/5551
A
HP → LE
LHC
AHA
Rücksprung
2923/5553
AMIN
HN − 1 → HN
2926/5556
DBLHN
DSP → AB
2927/5557
ADD
AB + KØ → AB
00 000 00L
2929/5561
DBSHN
AB → DSP
UTM
UTM
UTM
2928/5560
A
HP → LE
LHC
AHA
Rücksprung
2930/5562
A
HP → LE
SHC
AHA
Rücksprung

Rückfragen zu dieser Mappe, Berichtigungen und Erganzungen bitte an

CGK/TR 51
Herrn Wegner
Tel.: 4263

COMPUTER KONSTANZ

7750 Konstanz, Max-Stromeyer-Str. 116

Best.Nr.: 440.A2.G5

ÄNDERUNGSSTAND (ae)			Blatt 1
Reg. 5	ae	Auszutauschende bzw. zu ergänzende Seiten (·)	Datum
	-	-	10.7.76

Schnittstellen-Beschreibung

Kapitel: PE 4.3.4

1. Auflage vom 10.7.76

umfaßt Blatt: PE 4.3.4-1 bis 4.3.4-210

INHALTSVERZEICHNIS

1. SCHNITTSTELLEN-BESCHREIBUNG

1.1. Schnittstelle zum Rechner

Die Schnittstelle zum Rechner vom BMP besteht aus fünf Registern: DW, KA, KR, ST, SC und den Merkern STM, ZM, FM, SPM, BM, TFK.

Register DW

In das Register DW werden die Datenbytes für den Rechner geladen. Danach wird dem Kanal mit AF die Bereitstellung des Bytes gemeldet. Zuvor muß natürlich die Übernahmebereitschaft mit ZM gemeldet worden sein.

In der automatischen Datenphase (Schreiben) speichert die Hardware das zuletzt aufgeschriebene Byte in DR, das vorletzte in DW.

In der automatischen Datenphase (Lesen) wird das DW-Register mit den gelesenen Datenbytes überschrieben.

Register KA

Im Register KA übergibt der Kanal Informationsbytes. Zuvor muß allerdings mit STP quittiert worden sein. STP ist also die Durchschaltbedingung für KA.

Beim Schreiben in der automatischen Datenphase wird das KA-Register mit den zu schreibenden Informationsbytes überladen.

Register KR

Im Register KR werden die Meldungen für den Kanalverkehr übergeben. Es gilt folgende Bitzuordnung:

	7	6	5	4	3	2	1	0
KR:	STP	AF	EMP	GA	FE	IE	AN2	AN1

STP:	Startphase; hiermit erfolgt die Freigabe des KA-Registers.
AF:	Antwortfreigabe
EMP:	Empfangsbereit
GA:	GA-Schleife setzen
FE:	Fehler
IE:	Informationsende
AN2:	Anruf an Rechner 2
AN1:	Anruf an Rechner 1

Register ST

Das Register ST dient in der Hauptsache als Steuerregister für die Schreib-Lese-Elektronik. Die Bits 2 und 3 steuern die Koordinierungs-Hardware

ST3 = L, nach 400 ns = 0: Kanal 2 freigeben
ST2 = L, nach 400 ns = 0: Kanal 1 freigeben

Es können auch beide Bits = L/0 gegeben werden. Wenn sich nach der Kanalfreigabe die SC-Bits 4 und 5 ändern, lag Vorreservierung an (Anrufbedingung).

Register SC

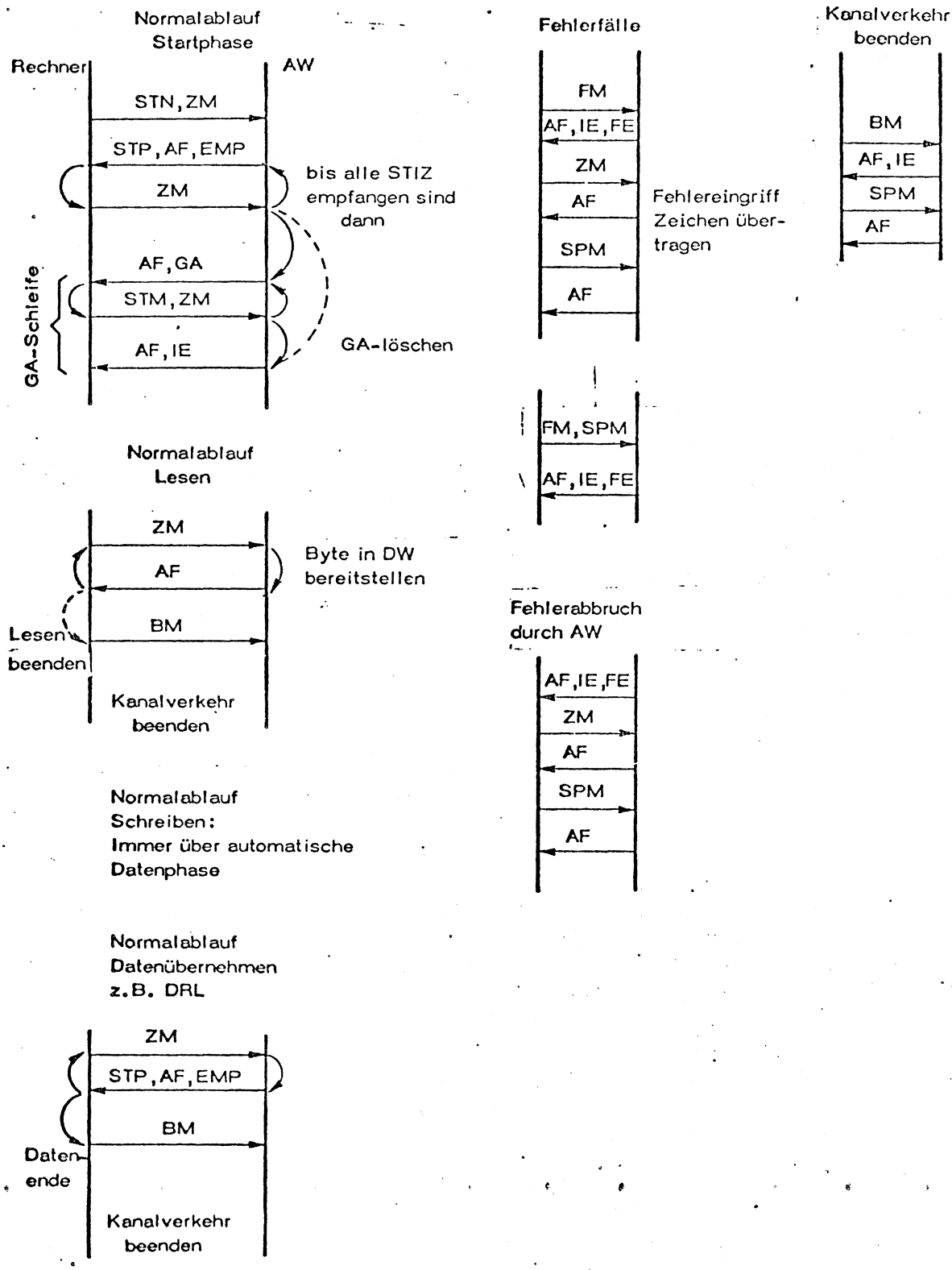
Im Register SC Bit 4 und 5 wird der Belegungszustand des AW gemeldet

SC4 = L (SC5 = 0): R1 reserviert das AW
SC5 = L (SC4 = 0): R2 reserviert das AW
SC4 = L, SC5 = L : es stand der Befehl "Lösche Reservierung" an. Nach Löschen der Reservierung (ST2, ST3 = L/0, L/0) kann nach frühestens 800 ns das SC-Register abgefragt werden. In Bit 4 bzw. 5 steht dann, welcher Kanal den Befehl gegeben hatte.

Merker

STM=L: Startmeldung vom Kanal liegt an
ZM =L: Der Kanal hat Zeichenmeldung gesendet, d.h., ist bereit zum Senden bzw. Empfangen eines Bytes.
FM =L: Fehlermeldung vom Kanal
SPM=L: Stopmeldung vom Kanal (Ende der Datenübertragung)
BM =L: Blockendemeldung vom Kanal
TFK =L: Rechner 1 verkehrt mit AW
=0: Rechner 2 verkehrt mit AW

Quittierungsfolge für Rechnerschnittstelle:



1.2. Schnittstelle zum Laufwerk

Diese Schnittstelle wird zur besseren Übersichtlichkeit in drei Teile aufgeteilt:

- Schnittstelle zur Schreib-Lese-Elektronik
- Schnittstelle zur ECC-Mimik
- Steuerung der Laufwerke

1.2.1. Schnittstelle zur Schreib-Lese-Elektronik

Diese Schnittstelle besteht aus vier Registern:

- Steuerregister ST,
- Datenlängenzähler Z2, Z1
- Datenregister DR
- Merker MD, M7

Steuerregister ST

Mit dem Steuerregister kann das Mikroprogramm die Schreib-Lese-Elektronik steuern. Die jeweilige Funktion muß mit ST = 0 zum richtigen Zeitpunkt abgebrochen werden.

Bedeutung der Bits in ST:

ST7 = L: ECC prüfen im Lesenfall (ID-Byte bis ECC-Bytes jeweils inclusive) nach Erkennung des nächsten Byteanfangs (MD)

ST6 = L: ECC-Bytes bilden im Schreibenfall (ID-Byte bis letztes Datenbyte)

ST6, ST7 = L, L: ECC-Bytes schreiben und vorletztes aufgeschriebenes Byte nach DW laden

ST5, ST7 = L, L: ECC-Bildung im Lesenfall abschalten (nach letztem ECC-Byte)

ST4 = L:

a) beim Lesen: Suche-Sync-Byte
Muß mindestens 14 µs vor Sync-Byte gegeben werden. Wenn Sync-Byte erkannt, setzt die Hardware MD. Ausnahme: Nach "AM-Search" TAG 11 BOB 6 s.S. 140 muß ST4 nicht gegeben werden; die Hardware sucht automatisch nach Sync-Byte, wenn eine AM erkannt wurde.

b) beim Schreiben: Pseudoschreiben (siehe ST1)

ST3: L/0 : Kanal 2 freigeben

ST2: L/0 : Kanal 1 freigeben

ST1: Pseudoschreiben, d.h., Schreibelektronik normieren. Bevor Schreiben zugeschaltet wird (TAG 11, FC2 (Write)), muß die Schreibelektronik normiert werden. Dazu muß die folgende Befehlsfolge gegeben werden:

- ST4 = L dann
- DR = 0 dann
- ST1 = L

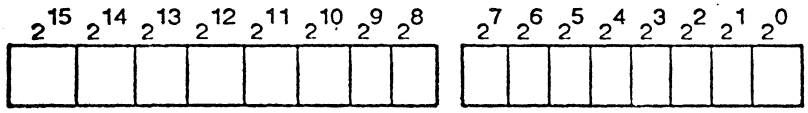
Danach wird Schreiben zugeschaltet. Vor Übergabe des 1. Bytes

- ST = 0

ST0 = L: GA-Schleife löschen

Datenlängenzähler Z2, Z1

Damit die Hardware in der automatischen Datenphase (M7) beim Lesen und Schreiben weiß, wieviel Datenbytes zu übertragen sind, muß das Mikroprogramm (MIP) zuvor den Datenlängenzähler laden.



Datenregister DR

Das Register DR ist die Datenverbindung zwischen AW und Laufwerk und zwar sowohl beim Lesen wie beim Schreiben.

Beim Lesen schreibt die Hardware das von der Platte gelesene Byte nach DR und setzt Merker MD. DR darf erst in dem Befehl nach der MD-Abfrage abgespeichert werden.

Beim Schreiben lädt das MIP das aufzuschreibende Byte nach DR. Hat die Hardware das Byte übernommen, setzt sie Merker MD.

Merker MD

MD wird von der Hardware gesetzt und muß von der Software gelöscht werden. Beim Lesen gibt der MD an, wann das Register DR mit einem Byte von der Platte beschrieben ist. Allerdings darf das Register DR erst in dem Befehl nach der MD-Abfrage abgespeichert oder abgefragt werden.

Beim Schreiben gibt der MD an, daß das Register DR von der Hardware übernommen ist.

Merker M7

M7 wird von der Software gesetzt. Damit beginnt die automatische Datenphase. Zuvor muß der Datenlängenzähler gesetzt werden und sichergestellt sein, daß ZM ansteht. Sind alle Bytes übertragen, löscht die Hardware den M7.

1.2.2.
Schnittstelle zur ECC-Mimik

Diese Schnittstelle besteht aus drei Registern: MB, SC, ME.

Register MB: ECC-Fehlererkennung 1 bei Lesen

Werden beim Prüfen der Information Fehler entdeckt, so meldet die Hardware diese Fehler in MB (teilweise auch an SC, siehe dort). Diese Meldungen sind für die ECC-Korrektur im MIP nötig. Die daraus resultierenden Reaktionen im MIP sind im Strukturdiagramm bzw. Flußdiagramm erläutert.

Im Folgenden werden nur die für das MIP interessanten Meldungen im MB beschrieben. Die übrigen Meldungen können der Hardwarebeschreibung * entnommen werden.

MB7 = L: Mindestens eine der Stellen 11-21 vom Polynomgenerator P0 ist ungleich 0.

MB4 = L: Die Stellen 0-10 von P1 sind ungleich denen von P0.

MB2 = L: Die Stellen 0-10 von P2 sind ungleich denen von P0.

MB0 = L: Die Stellen 0-10 von P3 sind ungleich denen von P0.

Register SC: ECC-Fehlererkennung 2 bei Lesen und Schreiben, Übertragungsfehler

Bestimmte Fehlerfälle in der ECC-Logik sowie bei der Datenübertragung werden im SC-Register gemeldet. Die Bedeutung der SC-Bits ist der Hardwarebeschreibung * zu entnehmen. Für das MIP bedeutet SC ≠ 0 immer den Abbruch der Datenübertragung. SC muß deshalb nach jeder Datenübertragung abgefragt werden.

Register ME

Die Stellen 11-18 des Polynomgenerators P0 werden auf die Stellen 7-0 des Registers ME geschaltet. Durch Shiften von P0 und anschließendes Abspeichern von ME wird im MIP das Fehlermuster aufbereitet.

*Siehe Beschreibung EAW 4031,PE 4.4 - Kapitel 6.6.1,6.6.3,6.6.4.

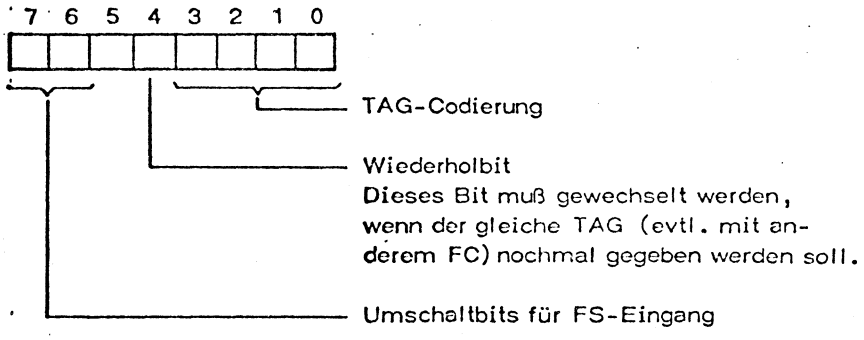
1.2.3.
Steuerung der Laufwerke

Diese Schnittstelle besteht aus den AW-Registern FT (TAG-Register), FC (Bus Out) und FS (Bus In) und dem Merker MI.

Der Merker MI wird gesetzt, wenn auf der Servospur der Index erkannt wird. Allerdings muß zuvor irgendwann SMI in MSL gegeben worden sein. Der Index steht 2µs lang an. Deswegen ist auch erst nach 2µs LMI wirksam.

In FT und FC stehen die Befehle an das Laufwerk, in FS stehen die Rückmeldungen des Laufwerks.

Bedeutung der Bits in FT:



Durch FT 7,6 wird ausgewählt, welche Meldungen in das FS-Register geladen wird. Es gilt folgende Zuordnung:

FT:	7	6
BUSIN → FS	0	0
Gerätefehler → FS FS0: Modul-Select-FE 1: Parity-FE BUSIN 2: Parity-FE BUSOUT 3: TAG Invalid 4: -- 5: Device Check	0	L
FS0-5: Mod.Adr.3/6 Code* FS6,7: AW-Adresse	L	0
DBUS-Zielregister-Freigabe	L	L

* Ist kein Modul angewählt, so besteht die 3/6-Code-Adresse aus 000 000.

Der Bus Out (FC) darf nicht geändert werden, ohne vorher erneut FT (mit geändertem Wiederholungsbit) gegeben zu haben. Einzige Ausnahme: TAG 11.

Bei den meisten TAG's muß ein Modul vorher adressiert werden. Ausnahmen sind hier:

TAG 2 POLL DEVICES, der die Interrupts aller Module abfragt und

TAG 3 TRANSMIT MODUL ADDRESS, der ja das Modul adressiert.

Zwischen dem Laden des FT-Registers und der Abfrage des FS (Bus In) müssen mindestens 4 µs liegen.

Wird nur der FS-Eingang umgeschaltet, nachdem schon mindestens 4 µs ein TAG anliegt, so genügt ein Befehl zwischen FT7,6 Umschaltung und der FS-Abfrage.

Tabelle 1, Seite 200 bzw. Tabelle 2, Seite 210 zeigt die Funktionen des TAG-Registers und des Bus Out und die Rückmeldungen auf dem Bus In. Im Folgenden werden diese Tabellen, geordnet nach der TAG-Numerierung, erläutert, soweit sie für das Mikroprogramm interessant sind; dabei wird auf die Unterschiede zwischen 100 und 200 MB-Version besonders hingewiesen.

TAG1 TRANSMIT SECTOR

Die Platte ist in 128 Winkel eingeteilt. Über den Bus Out wird der zu suchende Winkel an das Laufwerk übertragen und zwar ins Sector-Register. Gleichzeitig wird Record Search in Progress (siehe Bus In 0 von TAG 4 bzw. 9) gesetzt. Gelöscht wird dieses Bit erst mit "Interrupt reset", TAG 9 Bus Out Bit (BOB) 0. Mit jedem Winkelimpuls wird ein anderes Register, der Sector Counter um 1 erhöht. Stimmt Sector Counter mit Sector Register überein, sendet das Laufwerk einen Record Ready-Interrupt. Wird der Interrupt nicht gelöscht, bevor der Sector Counter weiter erhöht wird, so nimmt das Laufwerk den Interrupt weg, bis (nach einer Plattenumdrehung) der gesuchte Winkel wieder ansteht. Das wird solange wiederholt, bis der Interrupt gelöscht wird.

Auf dem Bus In wird der aktuelle Stand des Sector Counters gemeldet. Dabei wird jeder Winkel noch in zwei Hälften eingeteilt, deshalb die Meldung "High Side of Sector".

Da "Save Sector" TAG 11 BOB 0 das Sector-Register überschreibt, ist während der Winkelsuche "Save Sector" verboten.

TAG2 POLL DEVICES

Je nach gesetztem BOB werden hier entweder die Interrupts der Module 0-7 (BOB 0) oder der Interrupt des Service-Moduls (BOB 2) abgefragt.

Die Rückmeldungen auf dem Bus In sind:

Bus In Bit (BIB) 0: Modul 7 meldet Interrupt
1: Modul 6 meldet Interrupt usw.

TAG2 zerstört eine vorher bestehende Adressierung, d.h., nach TAG2 muß das Modul neu adressiert werden. TAG2 ist der einzige TAG, bei dem keine 3/6-Code-Adresse gemeldet wird, wohl aber die AW-Adresse im gleichen Byte.

Die Interrupt - Bedingungen werden mit "Interrupt reset" TAG 9 BOB0 gelöscht, außer "Seek Incomplete". Dieser Interrupt muß mit "Rezero Start" TAG 9 BOB 3 gelöscht werden.

Es gibt folgende Interrupt - Bedingungen:

- Seek Complete

  Der vorher gegebene Positionier-, Normier- oder Offset-Befehl (TAG 9 BOB 3, BOB 4, BOB 5 oder TAG 10 BOB 0) ist richtig beendet (d.h., On Cylinder und Diff. = 0 siehe TAG 12).

- Seek Incomplete

  Dieser Interrupt wird gegeben, wenn ein Positionieren nicht richtig beendet werden konnte. Folgende Gründe sind möglich:

  . Die Köpfe sind über Zylinder 410 hinausgefahren,
  . die Köpfe sind hinter Zylinder 0 zurückgefahren, ohne daß ein "Rezero Start" (TAG 9 BOB 3) gegeben wurde.
  . "On Cylinder" kommt nicht innerhalb 500 ms nach "Seek Start" oder "Rezero Start".

- Attention

  Aus zwei Gründen kann dieser Interrupt gegeben werden:

  . Das Laufwerk wurde eingeschaltet und ist BEREIT, z.B. nach Stapeltausch,
  . der Adreßstecker wurde gezogen und wieder eingesteckt.

- Record Ready

  Der gesuchte Winkel ist gefunden (siehe TAG 1).

TAG3 TRANSMIT MODUL ADDRESS

Die auf dem Bus Out angegebene Moduladresse wird mit den Adreßsteckern verglichen und dasjenige Laufwerk angewählt, welches den entsprechenden Stecker enthält.

Ab jetzt, d.h., solange ein Modul ausgewählt ist, sendet dieses Modul die 3/6-Code-Adresse an das AW.

Befinden sich zwei gleiche Adreßstecker in der Strecke, wird bei deren Adressierung "Modul-Select-Fehler" (FS0) gemeldet.

Auf dem Bus In wird bei 100 MB-Version nichts zurückgemeldet. Bei 200 MB-Version sind folgende Rückmeldungen möglich:

- 200 MB Drive (BIB1). Es handelt sich um ein 200 MB-Laufwerk.

Die folgenden Rückmeldungen sind nur möglich bei Laufwerken mit Doppelzugriff

- Reserved to this channel

  Das Laufwerk ist für den Zugreifer reserviert

- Reverserd to other channel

  Das Laufwerk ist für den anderen Zugreifer reserviert

- Engaged to other channel

  Das Laufwerk arbeitet mit dem anderen Zugreifer.

TAG4 REQUEST STATUS

TAG4 dient zum Abfragen des aktuellen Zustandes des Laufwerks. Der Bus Out ist irrelevant solange nicht gleichzeitig "Diagnostic Mode 2" ansteht (siehe Beschreibung weiter unten).

Folgende Rückmeldungen sind möglich:

- Index Error (BIB7):

  Die Laufwerklogik hat zwischen Winkel 127 und 0 das Indexmuster nicht erkannt. Diese Meldung wird bei der nächsten Indexerkennung weggenommen.

- Offset Active (BIB6):

  Bleibt gesetzt, solange ein Offset-Vorgang abläuft, bzw. die Köpfe in Offset-Position sind.

- Seek Incomplete (BIB5)
- Seek Complete (BIB4)
- Attention (BIB2)

  } siehe TAG2

  Diese drei Meldungen können nur über "Interrupt reset" TAG9 BOB0 weggenommen werden.

  Mit Attention kommt immer Seek Complete oder Seek Incomplete.

- On Line (BIB3):

  Sitzt immer, wenn das Laufwerk bereit ist.

- Busy (BIB1):

  Bleibt gesetzt, solange ein Positioniervorgang abläuft.

- Record Search in Progress (BIB0):

  Bleibt gesetzt ab TAG1 bis zu einem Interrupt reset (TAG9 BOB0), siehe auch TAG1.

Steht "Diagnostic Mode 2" (gesetzt durch TAG 13 BOB5) an, wenn TAG4 gegeben wird, werden durch den Bus Out die dort engegebenen Funktionen simuliert.

TAG5 REQUEST ADDRESS

Mit diesem TAG können mehrere Laufwerkregister je nach Bus Out auf den Bus In übertragen werden.

- Read Difference Counter (BOB4) bei 100 MB-Version bzw. Read Low Difference

  Hierdurch wird das Differenz-Register übertragen. Dies kann entweder die Differenz $2^0 - 2^7$ für das Positionieren sein oder die Offsetschrittweite inclusive Richtung (Offset Reverse).

- Read HAR (Head Address Register, BOB3)

  Das Kopfadreßregister wird übertragen.
  Darin enthalten ist bei 100 MBV Differenzbit 2^8, das Bit 2^8 der Zylinderadresse und die Richtung des Positioniervorganges (Reverse).

  Bei 200 MBV ist Bit 2^8 und 2^9 der Zylinderadresse enthalten.

- Read CAR (Cylinder Address Register, BOB 2)

  Das Zylinderadreßregister 2^0-2^7 wird übertragen.

- Read High Difference (BOB 1). Nur 200 MB-Version!

  Die Differenzbits 2^8 und 2^9 sowie die Positionierrichtung werden übertragen.

- Read Sector Register (BOB 0)

  Jeder der 128 Winkel ist in zwei Teile unterteilt, deswegen High Side of Sector.
  Um den aktuellen Winkel (vom Sector Counter) zu erfahren, muß "Save Sector" TAG 11 BOB 0 gegeben werden. Dies bewirkt, daß der Sector Counter in das Sector Register übernommen wird.

Mit "Read Sector Register" wird dann also der Winkel übertragen, der zum Zeitpunkt "Save Sector" anstand.
Zu beachten ist, daß es zwischen den einzelnen Winkeln einen Bereich (ca. 16 µs) gibt, in dem die Übernahme des Sector Counters verzögert wird, bis der neue Winkel ansteht, der dann auch übernommen wird. Aus diesem Grund muß zwischen "Save Sector" und "Read Sector Register" ca. 16 µs gewartet werden.

TAG6 TRANSMIT CYLINDER ADDRESS

Die im Bus Out angegebene (Ziel-) Zylinderadresse 2^0-2^7 wird im Zylinderadreßregister übernommen.
Außerdem wird das Differenzregister auf den höchsten Wert gesetzt, die Positionierrichtung wird auf vorwärts eingestellt und die Kopfadresse genullt.
Deswegen ist zum Positionieren die TAG-Reihenfolge TAG 5, TAG6, TAG7, TAG 8 (evtl. mehrmals TAG 5 zum Überprüfen), TAG 9 vorgeschrieben.
Auf dem Bus In wird die gerade eingeschriebene Adresse rückgemeldet. Damit ist eine Kontrolle möglich.

TAG 7 TRANSMIT HEAD ADDRESS

BOB 0-4 werden in das Kopfregister geschrieben.
Bei den 100 MB-Laufwerken wird die Positionierrichtung, das Zylinderadreßregister Bit 2^8 und das Differenzregister Bit 2^8 entsprechend den Angaben im Bus Out gesetzt. Auf dem Bus In wird nur die Kopfadresse zurückgemeldet.

Bei den 200 MB-Laufwerken werden die Zylinderbits 2^8 und 2^9 entsprechend den Angaben im Bus Out gesetzt. Auf dem Bus In wird das gerade eingeschriebene Byte zurückgemeldet.

TAG 8 TRANSMIT DIFFERENCE/TRANSMIT OFFSET

Das Differenzregister Bit 2^0 - 2^7 wird entsprechend Bus Out gesetzt. Wird später ein "Offset Start" TAG 9 BOB5 gegeben, so wird das Differenzregister als Offset-Schrittweite gewertet, BOB7 als Offset-Richtung. "Sign change" wird nicht ausgewertet.

Bei den 200 MB-Laufwerken ist als Offset-Schrittweite maximal 775µ inch angebbar gegenüber 1575µ inch bei den 100 MB-Versionen.

Auf dem Bus In wird die gerade geschriebene Differenz bzw. Offsetweite und -Richtung zurückgemeldet. Hier hat "Sign change" eine Bedeutung, wenn die Field Test Unit angeschlossen ist, siehe dort.

TAG9 TRANSMIT CONTROL I

Auf dem Bus In erscheinen die gleichen Meldungen wie bei TAG 4.
Folgende Funktionen können mit TAG 9 durchgeführt werden:

BOB6 METERING IN B

Diese Funktion wird nicht verwendet.

BOB5 OFFSET START

Gemäß den Bits im Differenzregister werden die Köpfe aus der Spurmitte gefahren. Die Beendigung des Offsetvorganges wird mit einem Seek Complete Interrupt gemeldet.

In Offset-Position kann gelesen aber nicht geschrieben werden. Bevor wieder positioniert werden soll, muß "Offset Reset" TAG 10 BOB0 erfolgen.

BOB4 SEEK START

Die Köpfe werden in der angegebenen Richtung um so viele Zylinder positioniert, wie dem Wert des Differenzregisters entspricht.

Diese Operation kann nicht durchgeführt werden, wenn die Köpfe in Offset-Position stehen. Ein zuvor gegebener "Seek Incomplete Interrupt" muß vor erneutem Positionieren gelöscht werden ("Rezero Start TAG9 BOB3).

BOB3 REZERO START

Die Köpfe werden auf Zylinder 0 zurückgefahren und das Zylinderadreßregister wird auf 0 gesetzt. Die Beendigung des Rezero wird mit "Seek Complete Interrupt" gemeldet.
Nach Eintreffen dieses Interrupts muß min. 33 ms gewartet werden, bevor der "Index Error" (mit TAG 4 oder TAG 9) abgefragt wird (logische Unsauberkeit in der Laufwerkslogik).

BOB 2 RESET HAR

Das Kopfadreßregister wird genullt.

BOB 1 CONTROL RESET

Es werden einige Zustände im Laufwerk gelöscht, hauptsächlich "Device Check".

BOB 0 INTERRUPT RESET

Die Interruptbedingungen werden gelöscht.

- Seek Complete
- Attention
- Record Search in Progress
- Sektor-Register

Die Seek Incomplete-Meldung wird nicht zurückgesetzt. Wenn Seek Incomplete gemeldet wird, muß vor dem nächsten Positionieren "Rezero Start" TAG 9 BOB 3 gegeben werden, da kein definierter Stand der Köpfe gewährleistet ist.

TAG 10 TRANSMIT CONTROL II

Auf dem Bus In erscheint keine Rückmeldung. Folgende Funktionen sind möglich:

BOB 4 RESET DIFFERENCE

Das Differenzregister wird auf den höchsten Wert (511) gesetzt.

BOB 1 DECREMENT DIFFERENCE

Das Differenzregister wird um 1 vermindert (hauptsächlich für Diagnostik-Zwecke).

BOB 0 OFFSET RESET

Die Köpfe werden wieder auf Spurmitte zurückgeführt. Dies muß auf einen "Offset-Start" folgen, bevor wieder geschrieben oder positioniert werden soll.

TAG 11 OPERATE

Der TAG 11 ist der einzige TAG, bei dem der Bus Out geändert werden darf, ohne erneut TAG 11 (Wiederholbit, siehe Seite 80) zu geben.

Im TAG 11 sind die Lese-Schreib-Operationen zusammengefaßt. Es müssen zum Teil Bitkombinationen und Zeitverhältnisse eingehalten werden. Dies ist u.a. im Folgenden für die einzelnen Fälle erläutert.

Zulässige BOB - Kombinationen

Write AM	AM search	Data enable	Head select	Head advance	Write	Read	Save Sector	
0	0	0	0	0	0	0	L	Sector-Counter → Sector-Register
0	0	0	L	0	0	0	0	Head select
0	0	0	L	0	L	0	0	Schreiben
L	0	0	L	0	L	0	0	AM schreiben
0	0	0	L	0	0	L	0	Lesen, nach 30 µs
0	0	L	L	0	0	L	0	Data enable zusätzlich
0	L	L	L	0	0	L	0	Adreßmarke suchen
0	0	0	0	L	0	0	0	Kopfadresse um 1 erhöhen

Vorgänge beim Schreiben

Vor jedem Schreibbefehl, d.h., vor Anlegen von "Write" BOB2, muß Pseudoschreiben gegeben werden (ST4, DR = 0, ST1 siehe Kapitel, Seite 40).

Bei jedem Lese-Schreib-Befehl muß zuerst der Kopf, dessen Adresse mit TAG7 übertragen wurde, mit "Head select" BOB4 ausgewählt werden. Nach frühestens 5 µs darf "Write" BOB2 zugeschaltet werden. Damit wird nun der Inhalt von DR auf die Platte geschrieben.

Beachte: Es darf nicht geschrieben werden, wenn die Köpfe in Offset-Position stehen.
Zum Schreiben der Adreßmarke muß für drei Bytes (3 . 1,24 µs) BOB7 angelegt werden.

Vorgänge beim Lesen

Zuerst wieder "Head select" BOB4. Frühestens nach 1 µs darf "Read" BOB1 angelegt werden. Nach 30 µs sollte "Read/Write Valid" BIB3 abgefragt werden und, wenn gesetzt "Data enable" BOB5 gegeben werden. Jetzt werden die Daten von der Platte ins DR-Register übertragen und können dort vom Programm abgeholt werden. Zu beachten ist hierbei, daß das DR-Register erst im Befehl nach der MD-Abfrage abgefragt werden darf.
Soll eine Adreßmarke gesucht werden, so muß "AM Search" BOB6 mindestens 5 µs vor dem Beginn der AM gegeben werden.

Sonstige Funktionen

"Head Advance" BOB3

Wenn ein ganzer Zylinder z.B. gelesen werden soll, kann das Hochzählen der Kopfadresse mit "Head advance" geschehen. "Head select", "Read" und "Write" dürfen nicht anliegen. Wird "Head advance" gegeben, wenn Kopf 18 ausgewählt ist, so wird "End of Cylinder" BIB1 und "Device-Check" (siehe Kap. 1.2.3, Seite 80) gemeldet. Device-Check muß mit "Control reset" gelöscht werden.

"Save Sector" BOB0

Durch "Save Sector" wird der aktuell anstehende Winkel aus dem Sector-Counter in das Sectorregister geschrieben und kann dort per TAG5 BOB0 gelesen werden. Wenn die Köpfe kurz vor dem nächsten Winkel stehen, wird die Übergabe verzögert, bis der neue Winkel ansteht und dieser wird dann in das Sectorregister geschrieben. Deswegen sollte zwischen "Save Sector" und "Read Sector-Register" (TAG5 BOB0) ca. 16 µs gewartet werden. Siehe auch Beschreibung TAG1 und TAG5.

Rückmeldungen

Auf dem Bus-In wird Folgendes gemeldet:

BIB 7 "Index Error"

Die Laufwerklogik hat zwischen Winkel 127 und 0 das Indexmuster nicht erkannt. Diese Meldung wird mit der nächsten Indexerkennung weggenommen.

BIB 6 "Offset Active"

Bleibt gesetzt, solange ein Offset-Vorgang abläuft bzw. die Köpfe in Offset-Position sind.

BIB 5 "Sign"

Nur für Arbeiten mit Field-Test-Unit interessant.

BIB 4 "Write Inhibited"

Der Schreibschutzschalter am Laufwerk ist gedrückt.

BIB 3 "Read/Write Valid"

Wird vom Laufwerk nach "Write" oder "Read" zurückgemeldet, wenn das Schreiben bzw. Lesen ordnungsgemäß abläuft.

BIB 2 "Index"

Steht nach der Indexerkennung 2 µs an.

BIB 1 "End of Cylinder"

Wird generiert, wenn ein Kopf mit der Adresse > 18 angesprochen werden soll, z. B. wenn bei Kopf 18 noch "Head advance" gegeben wird.

BIB 0 "Write Sense"

Dieses Bit wird zurückgemeldet, wenn zuvor "Write" BOB 2 gegeben wurde.

TAG 12 REQUEST DIAGNOSTIC SENSE und
TAG 13 MODE + DIAGNOSTIC CONTROL

dienen Diagnose-Zwecken und werden in diesem Zusammenhang nicht beschrieben.

TAG 14 RESERVE RELEASE

Nur 200 MB-Laufwerke mit Doppelzugriff.

BOB7 "Reserve"

Das adressierte Laufwerk wird für den Zugreifer reserviert.

BOB6 "Release"

Das adressierte Laufwerk wird wieder freigegeben.

Ob die angesprochene Funktion richtig ausgeführt wurde, kann mit TAG 3 überprüft werden.

TAG 15 TRANSMIT HIGH DIFFERENCE

Nur für 200 MB-Laufwerke

Die Positionierrichtung und die Differenzbits 2^8 und 2^9 werden entsprechend den Angaben im Bus Out gesetzt.
Auf dem Bus In wird das gerade eingeschriebene Byte zurückgemeldet.

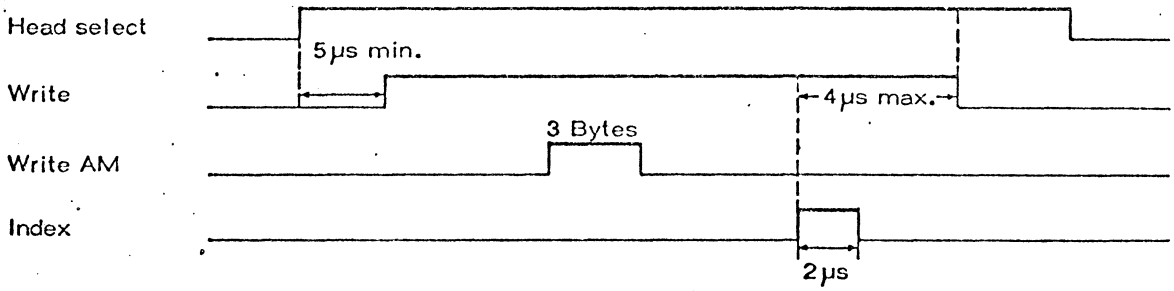
Zu beachtende Zeitverhältnisse:

1) Zwischen FT laden und FS abfragen: 4µs min.

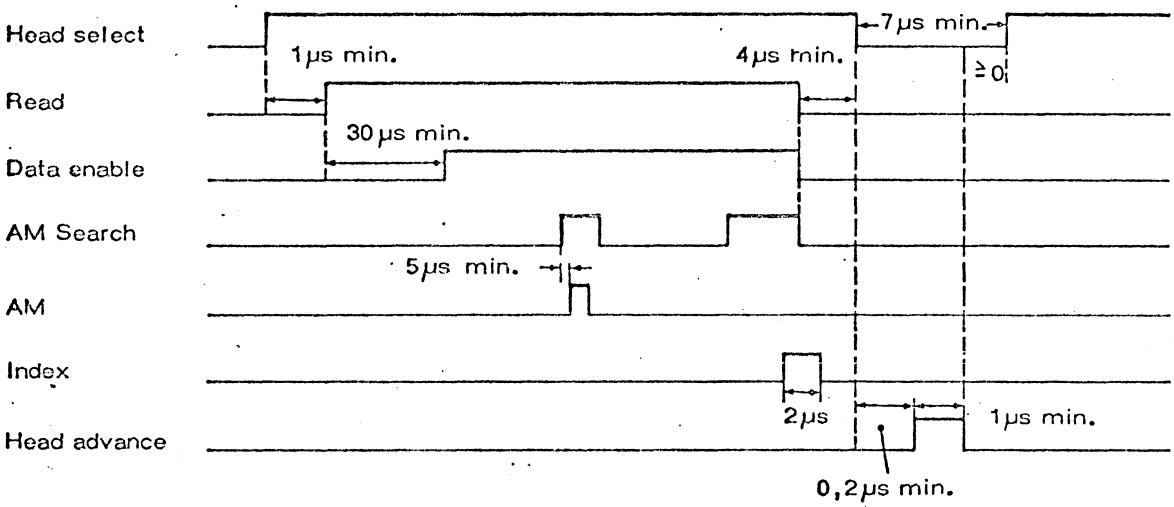
2) Nach Abschaltung FS-Eingang: 1 Befehl

3) Zwischen Save Sector und Read Sector Register: ca. 16µs min.

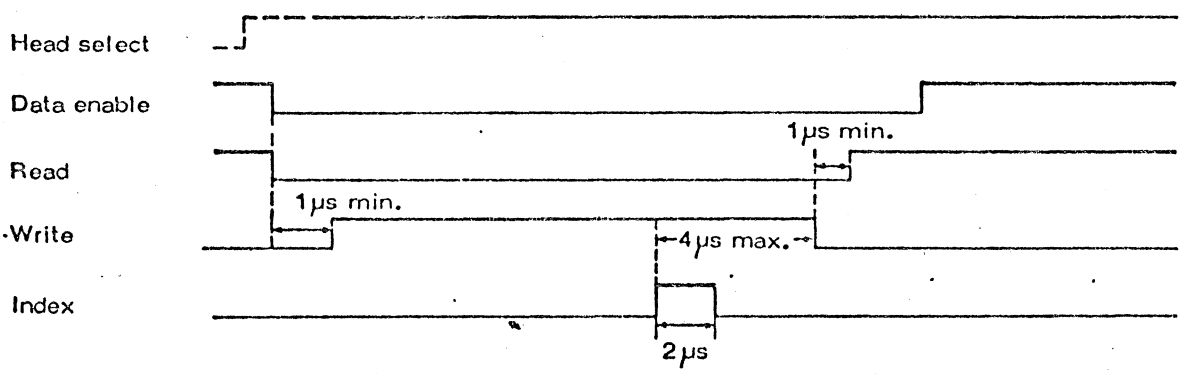
4) Zwischen ST4 und : 14µs min.



5) Schreiben zuschalten



6) Lesen zuschalten



7) Schreiben nach Lesen

BEI TAG BEDEUTET → BUS-OUT → UND BUS-IN

NAME	DE-CO-DE	BIT 7	6	5	4	3	2	1	0
TRANSMIT SECTOR	1		64	32	16	8	4	2	1
POLL DEVICES	2								NORMAL
							SERVICE		
TRANSMIT MODULE ADDRESS	3				SERVICE		4	2	1
REQUEST STATUS	4	SIMULATE EVEN DIBIT	SIMULATE FWD EOT ENABLE	SIMULATE REV EOT ENABLE	SIMULATE VELOCITY	BLANK ON CYL	SIMULATE FINE ENABLE	INHIB UNLD HDS SIMUL HDS LOADED	
REQUEST ADDRESS	5				READ DIFF CNTR				
						READ HAR			
							READ CAR		
									READ SECTOR REG
TRANSMIT CYLINDER ADDRESS	6	128	64	32	16	8	4	2	1
TRANSMIT HEAD ADDRESS	7	REVERSE	CAR 256	DIFF 256	16	8	4	2	1
TRANSMIT DIFFERENCE	8	128	64	32	16	8	4	2	1
TRANSMIT OFFSET		REVERSE	SIGN CHANGE	800 UIN	400 UIN	200 UIN	100 UIN	50 UIN	25 UIN
TRANSMIT CONTROL I	9		METERING IN B	OFFSET STAR"	SEEK START	REZERO START	RESET HAR	CONTROL RESET	INTERRUPT RESET
TRANSMIT CONTROL II	10				RESET DIFF			DECREMT. DIFFER-ENCE	OFFSET RESET
OPERATE	11	WRITE AM	AM SEARCH	DATA ENABLE	HEAD SELECT	HEAD ADVANCE	WRITE	READ	SAVE SECTOR
REQUEST DIAGNOS-TIC SENSE	12		SERVO POINTS						
				MONITOR MODE & DIAG					
					MONITOR STATE				
						CHECK STATUS			
							FAULT		
								TEST LOGIC	
MODE & DIAG CONTROL	13		DIAG MODE 4	DIAG MODE 2	DIAG MODE 1	BLOCK PARITY	MONITOR MODE 4	MONITOR MODE 2	MONITOR MODE 1

BIT 7	6	5	4	3	2	1	0
HIGH SIDE OF SECTOR	64	32	16	8	4	2	1
0	1	2	3	4	5	6	7
SERVICE							
INDEX ERROR	OFFSET ACTIVE	SEEK INCOM-PLETE	SEEK COM-PLETE	ON LINE	ATTEN-TION	BUSY	RECORD SEARCH IN PROGRESS
128/ OFFSET REVERSE	64	32	16	8	4	2	1
REVERSE	CAR 256	DIFF 256	16	8	4	2	1
128	64	32	16	8	4	2	1
HIGH SIDE OF SECTOR	64	32	16	8	4	2	1
128	64	32	16	8	4	2	1
			16	8	4	2	1
128	64	32	16	8	4	2	1
REVERSE	SIGN CHANGE	800 UIN	400 UIN	200 UIN	100 UIN	50 UIN	25 UIN
SAME BYTE AS TAG DECODE 4							
INDEX ERROR	OFFSET ACTIVE	SIGN	WRITE INHIBITED	READ/ WRITE VALID	INDEX	END OF CYLINDER	WRITE SENSE
$V_D > 64$	ON CYLINDER	FINE ANALOG	DIFF = 0	FINE MODE	DIBITS	CYLINDER PULSE	END OF TRAVEL
	DIAG MODE 4	DIAG MODE 2	DIAG MODE 1		MONITOR MODE 4	MONITOR MODE 2	MONITOR MODE 1
MONITOR LATCH 8	MONITOR LATCH 7	MONITOR LATCH 6	MONITOR LATCH 5	MONITOR LATCH 4	MONITOR LATCH 3	MONITOR LATCH 2	MONITOR LATCH 1
CE PGM STOP					MONITOR CHECK		COMMAND REJECT
SELECT LOCK	No SERVO TRACK	TEMP HIGH	NEG VOLT FAULT	POS VOLT FAULT		HEADS NOT LOADED	EVEN CYL
	DIAG MODE 4	DIAG MODE 2	DIAG MODE 1		MONITOR MODE 4	MONITOR MODE 2	MONITOR MODE 1

TAG-Tabelle für 100 MB-Laufwerk

PE

4.3.4 - 200

TAG-Tabelle für 200 MB-Laufwerke

BEI TAG BEDEUTET → BUS-OUT → UND BUS-IN

NAME	DECODE	BIT 7	6	5	4	3	2	1	0
TRANSMIT SECTOR	1		64	32	16	8	4	2	1
POLL DEVICES	2						SERVICE		NORMAL
TRANSMIT MODULE ADDRESS	3				SERVICE		4	2	1
REQUEST STATUS	4	SIMULATE EVEN DIGIT	SIMULATE FWD EOT ENABLE	SIMULATE REV EOT ENABLE	SIMULATE VELOCITY	BLANK ON CYL	SIMULATE FINE ENABLE	INHIB UNLD HDS SIMUL HDS LOADED	
REQUEST ADDRESS	5				READ DIFF CYL READ LOW DIFF	READ HAR	READ CAR	READ HIGH DIFF	READ SECTOR REG
TRANSMIT CYLINDER ADDRESS	6	128	64	32	16	8	4	2	1
TRANSMIT HEAD ADDRESS	7	REVERSE	CAR 256 CAR 512	DIFF 256 CAR 256	16	8	4	2	1
TRANSMIT DIFFERENCE	8	128	64	32	16	8	4	2	1
TRANSMIT OFFSET		REVERSE	SIGN CHANGE	800 UIN SIGN CHANGE	400 UIN	200 UIN	100 UIN	50 UIN	25 UIN
TRANSMIT CONTROL 1	9		METERING IN B	OFFSET START	SEEK START	REZERO START	RESET HAR	CONTROL RESET	INTERRUPT RESET
TRANSMIT CONTROL 2	10				RESET DIFF			DECREMENT DIFFERENCE	OFFSET RESET
OPERATE	11	WRITE AM	AM SEARCH	DATA ENABLE	HEAD SELECT	HEAD ADVANCE	WRITE	READ	SAVE SECTOR
REQUEST DIAGNOSTIC SENSE	12		SERVO POINTS	MONITOR MODE 4, DIAG	MONITOR STATE	CHECK STATUS	FAULT	TEST LOMC	
MODE & DIAG CONTROL	13		DIAG MODE 4	DIAG MODE 2	DIAG MODE 1	BLOCK PARITY	MONITOR MODE 4	MONITOR MODE 2	MONITOR MODE 1
RES. REL.	14	RESERVE	RELEASE						
TRANSMIT HIGH DIFF	15	REVERSE						DIFF 512	DIFF 256

BIT 7	6	5	4	3	2	1	0
HIGH SIDE OF SECTOR	64	32	16	8	4	2	1
0	1	2	3	4	5	6	7
SERVICE							
RSVD TO THIS CHANNEL	RSVD TO OTHER CHANNEL	ENGAGED TO OTHER CHANNEL				200 MB DRIVE	
INDEX ERROR	OFFSET ACTIVE	SEEK INCOMPLETE	SEEK COMPLETE	ON LINE	ATTENTION	BUSY	RECORD SEARCH IN PROGRESS
128/OFFSET REVERSE	64	32	16	8	4	2	1
REVERSE	CAR 256 CAR 512	DIFF 256 CAR 256	16	8	4	2	1
128	64	32	16	8	4	2	1
REVERSE						DIFF 512	DIFF 256
HIGH SIDE OF SECTOR	64	32	16	8	4	2	1
128	64	32	16	8	4	2	1
REVERSE	CAR 256 CAR 512	DIFF 256 CAR 256	16	8	4	2	1
128	64	32	16	8	4	2	1
REVERSE	SIGN CHANGE	800 UIN SIGN CHANGE	400 UIN	200 UIN	100 UIN	50 UIN	25 UIN
SAME BYTE AS TAG DECODE 4							
INDEX ERROR	OFFSET ACTIVE	SION	WRITE INHIBITED	READ/WRITE VALID	INDEX	END OF CYLIND LOC/PAD OPER	WRITE SENSE
$V_D > 56$ $V_D > 128$	ON CYLINDER	FINE ANALOG	DIFF = 0	FINE MODE	DIBITS	CYLINDER PULSE	END OF TRAVEL
	DIAG MODE 4	DIAG MODE 2	DIAG MODE 1		MONITOR MODE 4	MONITOR MODE 2	MONITOR MODE 1
MONITOR LATCH 8	MONITOR LATCH 7	MONITOR LATCH 6	MONITOR LATCH 5	MONITOR LATCH 4	MONITOR LATCH 3	MONITOR LATCH 2	MONITOR LATCH 1
CE PGM STOP	MOTOR OFF	MULTIPLE HEAD FAULT	CURRENT FAULT	INTERFACE CHECK	MONITOR CHECK	INTERLK NOT COMPLETE	COMMAND REJECT
SELECT LOCK	NO SERVO TRACK	TEMP HIGH AIR FLOW FAULT	NEG VOLT FAULT PAD UNSAFE	POS VOLT FAULT VOLT FAULT	WRITE OVERRUN	HEADS NOT LOADED	EVEN CYL
	DIAG MODE 4	DIAG MODE 2	DIAG MODE 1		MONITOR MODE 4	MONITOR MODE 2	MONITOR MODE 1
REVERSE						DIFF 512	DIFF 256

252

Schnittkanten

▭ = 200 M Byte ▭ = 2 Anpaßwerke an 1 LW

▭ = grün ▭ = rot

300

Oberfläche gebürstet, frei von Riefen u Kratzern, eloxiert
Grund Alu-matt
Schrift, Linien u Innenrandline sowie Pfeile schwarz RAL 9005
verschiedene Felder grün bzw rot.

Rückseite selbstklebend
Schrift: Helvetica Versalien 2 bzw. 5 mm
Abdeckfolie Alu-farbig n Wahl d. Herst.
Lieferer: Metall-Atzwerk, Schaidegg o.ändere

PE 4.3.4 - 210

ÄNDERUNGSSTAND (ae)			Blatt 1
Reg. 4	ae	Auszutauschende bzw. zu ergänzende Seiten (·)	Datum
	-	-	10.7.76

ECC-Verfahren

Kapitel: PE 4.3.5

1. Auflage vom 10.7.76

umfaßt Blatt: PE 4.3.5-1 bis 4.3.5-150

INHALTSVERZEICHNIS

1. ECC-VERFAHREN

Erläuterung des Fehlerkorrekturverfahrens für Wechselplattenspeicher (PST 116-3, AW WSP 430, IBM Spurformat 3830/3330) [M. Prögler, J. Swoboda, AEG-TELEFUNKEN Forschungsinstitut].

1.1. Allgemeines

Das Verfahren basiert auf einem Vorschlag von R.T. Chien (Burst-Correcting Codes with High-Speed Decoding, IEEE Trans. Inf. Theory, vol. IT - 15, Jan. 1969, S. 109-113). Für die Realisierung wurde es leicht modifiziert.

1.2. Codieren

Zum Schutz gegen Fehler werden an einen Block von $k \leq k_o$ Datenbits (entsprechend $k = 8 . <$ Zahl der Datenbytes $>$) m Redundanz- oder Prüfbits (hier $m = 8 . 7 = 56$ Bits) abgeleitet und dem Block der Datenbits angefügt. Damit entsteht ein Codeblock von insgesamt $n = k + m$ Bits. Die Prüfbits werden aus den Datenbits mit Hilfe eines rückgekoppelten Schieberegisters gewonnen. Die Periode n_o des Schieberegisters begrenzt die Länge des Codeblocks: $n_o = k_o + m \geq k + m$ (hier: $n_o = 585\,442$, siehe Seite 30).

Das Register mit seinen Rückkopplungen kann einem sog. Generatorpolynom P(x) zugeordnet werden, wobei ein Koeffizient $p_i = 1$ einem vorhandenen und $p_i = 0$ einem fehlenden Rückkoppelweg entspricht.

$$P(x) = p_m \cdot x^m + \ldots + p_1 \cdot x^1 + p_o$$

Die Folge der k Datenbits $d_{k-1}, d_{k-2}, \ldots d_1, d_o$ wird ebenfalls als ein Polynom aufgefaßt. Dieses Datenpolynom ist

$$D(x) = d_{k-1} \cdot x^{k-1} + d_{k-2} \cdot x^{k-2} + \ldots + d_1 \cdot x + d_o$$

Bei serieller Übertragung läuft die Reihenfolge mit fallendem Index bzw. fallender Potenz von x.

In gleicher Weise wird die Folge der Prüfbits einem Redundanzpolynom zugeordnet:

$$R(x) = r_{m-1} \cdot x^{m-1} + r_{m-2} \cdot x^{m-2} + \ldots + r_1 \cdot x + r_o$$

Mathematisch ergibt sich das Redundanzpolynom $R(x)$, indem das Datenpolynom $D(x)$ mit x^m multipliziert wird, und dann der Rest bezüglich des Generatorpolynoms $P(x)$ gebildet wird. Bei dieser Rechnung sind die Koeffizienten modulo 2 zu addieren.

$$R(x) = \text{Rest}\ \frac{D(x) \cdot x^m}{P(x)}$$

Das Codepolynom (Codeblock) entsteht durch Addition von $D(x)x^m$ und $R(x)$:

$$C(x) = D(x)x^m + R(x) = C_{n-1}x^{n-1} + \ldots + C_1x + C_o$$

(Das ist ein einfaches "Aneinanderfügen" von Daten- und Prüfblock, da $D(x)x^m$ in den letzten m Stellen nur Nullen hat.) Das Codepolynom $C(x)$ ist durch das Generatorpolynom $P(x)$ ohne Rest teilbar, da in $D(x) \cdot x^m$ auch $R(x)$ additiv enthalten ist. $R(x) + R(x)$ ergibt aber binär addiert = 0.

Das folgende Codierregister kann die Multiplikation mit x^m mit der Restbildung bezüglich $P(x)$ ausführen. Die Restbildung wird durch die Rückkoppelwege bewirkt, und die Multiplikation mit x_m wird durch die Einspeisestelle am Rückkoppelweg p_m für die Folge der Datenbits bewirkt. Während der Eingabe von $D(x)$ sind die Rückkopplungen wirksam; gleichzeitig wird $D(x)$ in den Speicher gegeben. Nach insgesamt k Schritten steht im Register die Redundanz $R(x)$, welche nun bei unwirksamer Rückkopplung ausgegeben und ebenfalls - den Daten nachfolgend - abgespeichert wird.

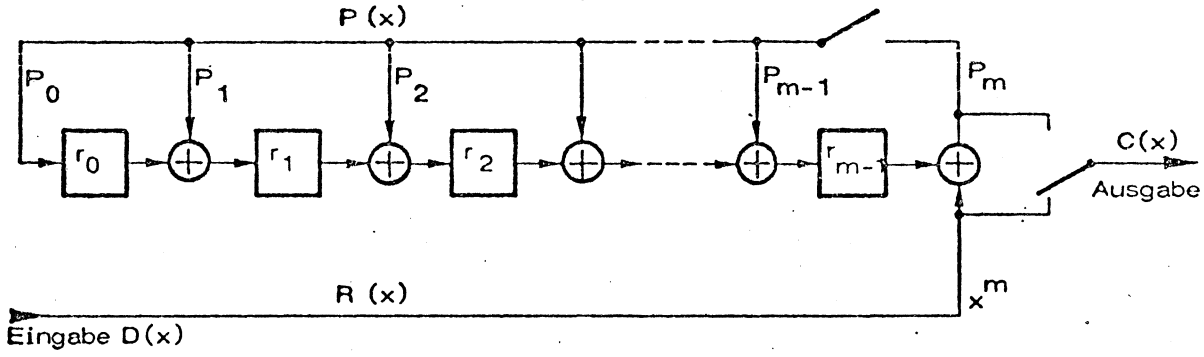


Abb. 1 Kodierregister mit m Stufen

1.3.
Code

Das Generatorpolynom P(x) besteht aus insgesamt vier Polynomen:

$$P_0(x) = x^{22} + 1$$
$$P_1(x) = x^{11} + x^7 + x^6 + x + 1$$
$$P_2(x) = x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$$
$$P_3(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1$$

Durch Ausmultiplizieren ergibt sich daraus das Generatorpolynom:

$$P(x) = P_0(x) \cdot P_1(x) \cdot P_2(x) \cdot P_3(x)$$
$$= x^{56} + x^{55} + x^{49} + x^{45} + x^{41} + x^{39} + x^{38} + x^{37} + x^{36} + x^{31} + x^{22} +$$
$$x^{19} + x^{17} + x^{16} + x^{15} + x^{14} + x^{12} + x^{11} + x^9 + x^5 + x + 1$$

Dieses Polynom ist beim "SCHREIBEN" wirksam und entsprechend Abb. 1 realisiert.

Für die Periode (Ordnung) e_i der einzelnen Polynome $P_i(x)$ gilt:

$\underline{e_0} = \underline{22}$

$\underline{e_1} = 2047/ggT(2047,299) = 23 \cdot 89/23 = \underline{89}$

$\underline{e_2} = 4095/ggT(4095,315) = 3 \cdot 3 \cdot 5 \cdot 7 \cdot 13/3 \cdot 3 \cdot 5 \cdot 7 = \underline{13}$

$\underline{e_3} = 2047/ggT(2047, 89) = 23 \cdot 89/89 = \underline{23}$

(ggT = größter gemeinsamer Teiler)

Daraus erhält man für die natürliche (unverkürzte) Blocklänge n_o des Codes, also für die Periode des SCHREIB-Registers mit dem Polynom P(x):

$\underline{n_o} = kgV(e_o, e_1, e_2, e_3) = kgV(22, 89, 13, 23) = 585\,442 =$ sedezimal '8EEE2'

(kgV = kleinstes gemeinsames Vielfaches)

Dies ist die "MODULUS-KONSTANTE" im Flußdiagramm.

Die maximal zu schützende Information kann also

$k_o = 585\,442 - 56 = 585\,386$ Bits bzw. 73 173 Bytes umfassen.

(Für $k < k_o$ kann man sich die führenden $k_o - k$ Informationsstellen als Nullen denken, die unterdrückt werden können, da sie den Inhalt des Codierregisters nicht verändern.)

Allgemein hat der Generator bei dem diskutierten Verfahren P(x) die Form

$$P(x) = (x^c + 1) \prod_{i=1}^{j} P_i(x)$$

Die $P_i(x)$ sind irreduzibel, haben den Grad g_i und die Ordnung (Periode) e_i, wobei kein e_i in c enthalten ist $(c = e_o)$. Der Code entdeckt alle Bündel bis zur Länge d und korrigiert gleichzeitig alle Bündel bis zur Länge l, die als Polynom aufgefaßt relativ prim zu $\prod_{1}^{j} P_i(x)$ sind. Weiterhin muß gelten:

$$c \geq l + d - 1 \text{ und } l \leq \sum_{1}^{j} g_i$$

Mit Sicherheit sind also alle die Bündel korrigierbar, deren Länge nicht größer als der kleinste Grad, $\text{Min}(g_i)$, $i = 1$ bis j, aller Polynome $P_i(x)$ ist. Im speziellen Fall sind das dann Bündel bis zur Länge $l = 11$, die sicher korrigierbar sind.

1.4. Decodieren

1.4.1. Allgemeines

Das Codewort bzw. dessen Datenteil kann durch ein Fehlerbündel verfälscht werden. Dieses Bündel habe die Länge l und kann ebenfalls als Polynom dargestellt werden:

$$B(x) = b_{l-1}x^{l-1} + \ldots + b_1 x + b_o$$

Darin bedeutet $b_j = 1$ eine Fehlerstelle und $b_j = 0$ eine ungefälschte Stelle.

Das gefälschte Codewort C(x) wird dargestellt durch Addition (mod. 2) von B(x) an der entsprechenden Stelle von C(x):

$$C'(x) = C(x) + x^i B(x) \quad 0 \leq i \leq n-1$$

Der Faktor x^i "rückt das Bündel an die richtige Stelle". Das Bündel von b_{l-1} bis b_o erstreckt sich dann im Codewort über die Stellen mit der Wertigkeit von x^{i+b-1} bis x^i; d.h., über die Stellen von i + b - 1 bis i, wenn man die Codewortstellen von hinten mit 0 beginnend zählt.

Bei der Decodierung gilt es festzustellen:

1. Ist der Block fehlerhaft?
2. Wenn ja, ist der Block korrigierbar?
3. Wenn ja, wie lautet B(x) und x^i? (Welche Konfiguration hat das Bündel und wo sitzt es?)

Alle diese Antworten können aus dem sog. Syndrom S(x) ermittelt werden. Dies ist der Rest, der bei der Division von C'(x) durch P(x) bleibt. Wegen der Linearität von Codierung und Fehleraddition gilt:

$$S(x) = \text{Rest}\,\frac{C'(x)}{P(x)} = \text{Rest}\,\frac{C(x)}{P(x)} + \text{Rest}\,\frac{x^i B(x)}{P(x)} = 0 + \text{Rest}\,\frac{x^i B(x)}{P(x)}$$

$$S(x) = s_{m-1}x^{m-1} + \ldots + s_1 x + s_o$$

Das Syndrom S(x) stellt also den Rest dar, der bei der Division des Fehlerpolynoms durch den Generator bleibt.

Das Register zur Syndromberechnung ist - bis auf die Einspeisestelle für C'(x) - mit dem Codierregister identisch (s. Abb. 2). Die Einspeisestelle am Rückkoppelweg p_o entspricht einer Vormultiplikation mit $x^o = 1$, d.h., der entfallenden Vormultiplikation in der obigen Formel für S(x).

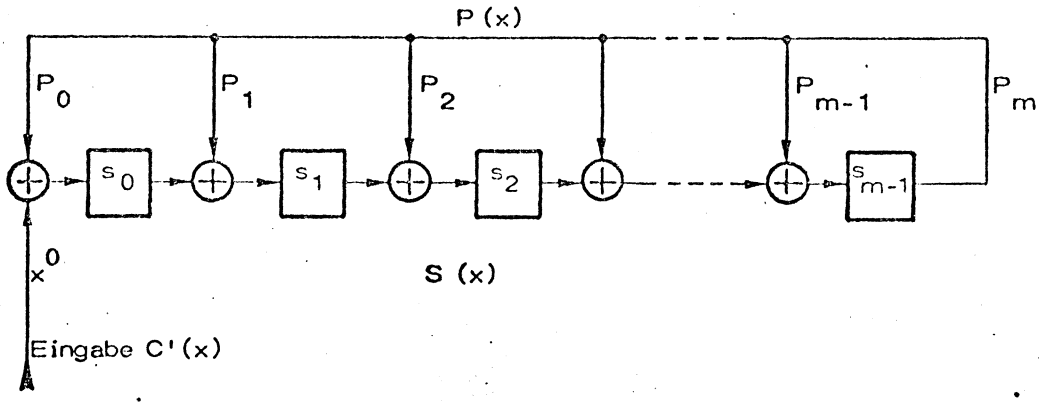


Abb. 2 Syndromregister

Das Syndrom steht im Register nach Eingabe aller n Bits von C'(x).

Das Syndrom hat nur dann den Wert Null, wenn entweder kein Fehler aufgetreten ist oder wenn das Fehlerbündel selbst ein erlaubtes Codewort ist, also nicht erkennbar ist.

Zur Ortung des Bündels genügt es, das Syndromregister mit wirksamer Rückkopplung, jedoch ohne Eingabe, weiter zu betreiben (Korrekturphase). Ein Bündel der Länge l steht dann in den unteren ("linken") l Stufen des Registers, wenn die oberen ("rechten") m - l Stufen alle Null enthalten. Ein korrigierbares Bündel muß innerhalb von n_o Schritten, der Periode des Registers, gefunden sein, sonst ist es erkennbar, jedoch nicht korrigierbar. Die Zahl s_i der Schritte bestimmt den Faktor x^i und damit den Bündelort:

$$i = n_o - s_i = -s_i \pmod{n_o}$$

Zur Korrektur wird der Fehler $x^i B(x)$ dem fehlerhaften Block C'(x) hinzuaddiert.

Das Verfahren besticht durch seine Einfachheit. Ein offensichtlicher Nachteil ist jedoch, daß die Zahl der Suchschritte im Extremfall gleich der Blocklänge n_o des Codes werden kann. Hier liegt der große Vorteil der Methode von Chien.

1.4.2.
Verfahren von Chien

Dieses Verfahren läßt sich dann vorteilhaft anwenden, wenn sich das Generatorpolynom P(x) als Produkt von Polynomen darstellen läßt. Das Codewort C(x) ist dann durch jedes Teilpolynom ohne Rest teilbar. Als Verallgemeinerung der von FIRE angegebenen Codes hat P(x) dann die Form:

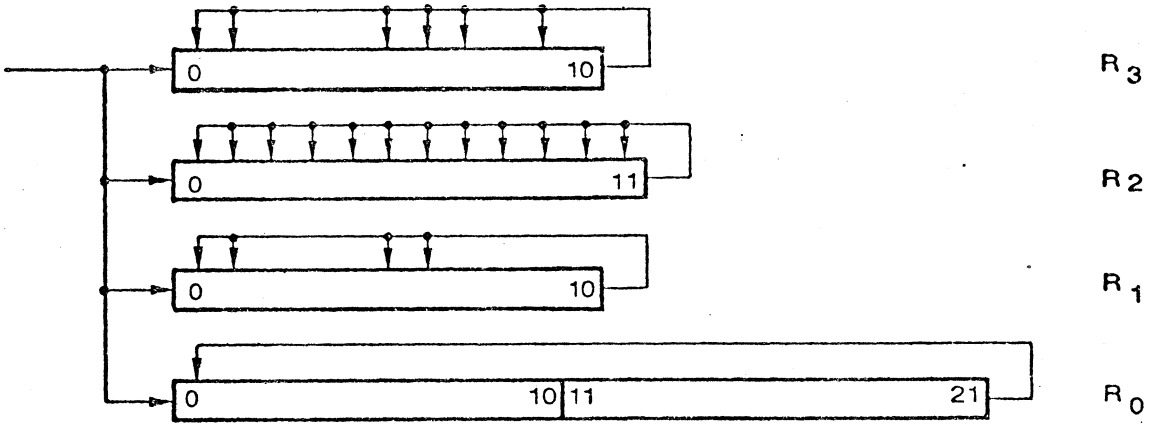
$$P(x) = \underbrace{(x^c + 1)}_{P_o(x)} \prod_{r=1}^{j} P_r(x)$$

Der Decoder besteht nicht aus einem, sondern aus $j + 1$ Syndromregistern R_o bis R_j, im speziellen Fall also aus insgesamt $j + 1 = 4$ Registern.

Das Register R_o entspricht dem Teilgenerator $P_o(x)$ und dient zuerst zur Ermittlung der Bündelkonfiguration B(x). Da R_o nur eine Rückkopplung besitzt, enthält R_o bei der Division von C'(x) durch $P_o(x)$ das Bündel unverändert an irgend einer Stelle. Weiterhin dient R_o sowie die übrigen Register R_1 bis R_j zur Bestimmung des Bündelortes ("DISPLACEMENT").

Das Decodieren beginnt mit der vollständigen Eingabe des Blockes C'(x) gleichzeitig in alle Register. Wie oben gezeigt wurde, ergeben sich dabei die gleichen Registerinhalte, die sich bei der Eingabe des Fehlerpolynoms $B(x) \cdot x^i$ allein ergeben würden. Mit Hilfe dieser Erkenntnis ist das Decodierverfahren leicht und anschaulich verständlich:

Wir nehmen an, daß in die $j + 1 = 4$ Register R_o bis R_3 mit dem Ausgangszustand Null nur Nullen und dann ein Fehlerbündel von maximal $l = 11$ Stellen eingespeist werden. Dann enthalten die vier Register unmittelbar nach Einlauf der letzten Fehlerstelle die Fehlerbündel in unveränderter Form, da die Rückkopplungen bisher nur Nullen zurückgekoppelt haben. Die Fehlerstellen b_{10}, b_9, b_8 ... b_1, b_0 stehen dann in allen vier Registern in den Stellen 10, 9, 8 ... 1, 0.



Bei dem Fehlerpolynom $B(x) \cdot x^i$ folgen auf die letzte Bündelstelle noch i Nullen nach, die ebenfalls in alle vier Register eingespeist werden. Bei dem Nachschieben der Nullen erscheint das Bündel in den Stellen 10, 9 ... 1, 0 des Registers R_r $(r = 0, 1, 2, 3)$ nach jeweils der Periode des Registers von e_r $(r = 0, 1, 2, 3)$ Takten.

Wenn das Bündel sich auf die zeitlich letzten $l = 11$ Stellen des Codeblocks erstreckt - das Bündelende liegt dann in der Zählweise von hinten beginnend in Stelle $i = 0$ - dann steht das Bündel am Ende der Einlaufphase von $C'(x)$ bzw. $B(x) \cdot x^{o}$ in den Stellen 10, 9 ... 1, 0 aller vier Register. Das Register R_o enthält dann in den Stellen 11 bis 21 nur Nullen. Wenn umgekehrt $R_{o(11-21)}$ nur Nullen enthält, dann kann das Bündelende zwar auf $i = 0$ gelegen haben, aber ebenso kann das Bündelende auf $i = 22, 44 \ldots$, also auf einem Vielfachen der Periode $c_o = 22$ gelegen haben.

Wenn das Register dagegen in der Korrekturphase erst um s_o Stellen zyklisch verschoben werden muß, um den Nulltest für $R_{o(11-22)}$ zu erreichen, dann kann das Bündelende in $i = 22 - s_o$ oder $i = 44 - s_o$ oder ... gelegen haben. Die Zahl der Schiebeschritte s_o legt den Ort des Bündelendes somit auf Vielfache von $e_o = 22$ fest.

$$i = -s_o \mod 22$$

Für die anderen Register R_1, R_2, R_3 gilt Ähnliches. Bei Einspeisen eines Fehlerbündels $B(x) \cdot x^{i}$, das in Stelle i endet, enthält das Register R_r $(r = 1, 2, 3)$ das unveränderte Bündel nach Einlauf von Stelle i, dann nach der Periode e_r wieder nach Einlauf von Stelle $i - e_r$, nach $i - 2 \cdot e_r$ usw., d.h., wenn das Nachschieben von i Nullen in der Einlaufphase mittels s_r Schiebetakte in der Korrekturphase fortgesetzt wird, wobei i durch s_r auf ein Vielfaches der Periode e_r ergänzt wird $(i + s_r = 0 \mod e_r)$, dann enthält das Register wieder das Fehlerbündel in unveränderter Form. Wann dies erfüllt ist, zeigt ein Vergleich von R_r mit $R_{o(0-10)}$, das bei erfülltem Nulltest $R_{o(11-21)}$ dieses Fehlerbündel auch enthält. Die Zahl s_r der Schiebeschritte bis zum erfüllten Vergleich legt den Ort des Bündelendes somit auf Vielfache von e_r fest.

$$i = -s_r \mod e_r$$

In dem diskutierten Verfahren muß der Ort des Bündelendes aus den folgenden vier Gleichungen bestimmt werden.

$$\begin{aligned} i &= -s_o \mod 22 \\ i &= -s_1 \mod 89 \\ i &= -s_2 \mod 13 \\ i &= -s_3 \mod 23 \end{aligned}$$

Ehe gezeigt wird, wie sich i aus den Shiftzahlen berechnen läßt, wird noch zusammengefaßt, was beim Ermitteln der Shiftzahlen zu beachten ist. Das Decodieren beginnt mit der vollständigen Eingabe des Blockes $C'(x)$ gleichzeitig in alle Register. Ist dann der Inhalt aller Register Null, dann liegt kein oder ein nicht erkennbarer Fehler vor. Enthalten nur einige (jedoch nicht alle) Register den Wert Null, so ist der Fehler erkennbar, jedoch nicht korrigierbar.

Als erstes wird während der Korrekturphase das Register R_o solange zyklisch verschoben, bis die oberen $c - l$ ($= 22 - 11 = 11$ in unserem Fall) Stellen alle Nullen enthalten; dann steht das Bündel selbst in den unteren l Stellen. Die dafür benötigte Zahl von Schritten ist s_o. Nun werden alle übrigen Register (mit wirksamer Rückkopplung) solange betätigt (das kann gleichzeitig erfolgen), bis jedes das Bündel in den untersten l Stellen enthält. Die Feststellung erfolgt einfach durch Vergleich der Inhalte mit dem von R_o. Ist das Bündel korrigierbar, dann muß es schließlich in allen j Registern direkt stehen, und zwar im Register R_r nach spätestens e_r (der Periode) Schritten. Im allgemeinen ist dies nach jeweils s_r Schritten der Fall, wobei also gilt: $s_r \leq e_r$. Man sieht, daß bei simultanem Betrieb aller Register die maximale Zahl von Suchschriften gegeben ist mit:

$$s_{max} = e_o + Max(e_r) \quad r = 1 \text{ bis } j$$

Dies ist oft Größenordnungen kleiner als n_o!

Für den gegebenen Code beträgt die maximale Zahl von Suchschritten:

$$s = 22 + 89 = 111$$

was nur etwa 1/5000 von n_o ist!

1.4.2.1.
Ermittlung des Ortes i für das Bündelende aus den Shiftzahlen

Aufgrund des "chinesischen Restsatzes" ist eine Zahl i durch ihre Reste

$$i \bmod e_r \qquad r = 0, 1, 2 \ldots j$$

bis auf ein Vielfaches von $n_o = kgV(e_o, e_1 \ldots e_j)$ eindeutig bestimmt.

Die Zahl i läßt sich berechnen mittels

$$i = \sum_o^j A_r \cdot (i \bmod e_r) \bmod n_o$$

Dabei sind A_r $(r = 0, 1 \ldots j)$ Konstanten. In unserem Fall sind die Shiftzahlen s_r $(r = 0, 1 \ldots j)$ bis auf das Vorzeichen diese Reste

$$s_r = (-i) \bmod e_r \qquad r = 0, 1 \ldots j$$

Damit stellt sich i aus den Shiftzahlen

$$i = -\sum_0^j A_r \cdot s_r \bmod n_o$$

Im folgenden sollen die Konstanten für den diskutierten Fall ermittelt werden.

Aus
$$i = (i \bmod 22) \cdot A_o \bmod n_o$$
$$+(i \bmod 89) \cdot A_1$$
$$+(i \bmod 13) \cdot A_2$$
$$+(i \bmod 23) \cdot A_3$$

läßt sich z. B. die Konstante A_o ermitteln, indem man für i ein Vielfaches (a) von 89 . 13 . 23 einsetzt. Auf der rechten Gleichungsseite fallen dabei die drei letzten Summanden weg.

$$a \cdot 89 \cdot 13 \cdot 23 = (a \cdot 89 \cdot 13 \cdot 23 \bmod 22) \cdot A_o \bmod n_o$$

Für a = 1 ergibt die Klammer den Rest 13. Es wird a schrittweise erhöht, bis sich der Rest 1 ergibt.

a	a . 89 . 13 . 23 mod 22
1	13
2	13 + 13 = 26 = 4 mod 22
3	4 + 13 = 17
4	17 + 13 = 30 = 8 mod 22
⋮	
16	19 + 13 = 32 = 10 mod 22
17	10 + 13 = 23 = 1 mod 22

Damit lautet obige Gleichung

$$\underbrace{17 \cdot 89 \cdot 13 \cdot 23}_{= 452\,387} = \underbrace{(17 \cdot 89 \cdot 13 \cdot 23 \bmod 22)}_{= 1} \cdot A_o \bmod n_o$$

bzw.

$$A_o = 452\,387$$

PE

Für die übrigen Konstanten ergibt sich nach entsprechender Rechnung:

$$A_1 = 72358$$
$$A_2 = 315238$$
$$A_3 = 330902$$

Es gilt außerdem noch

$$\sum_{0}^{3} A_i = 1 \bmod n_o$$

Dieser Zusammenhang ergibt sich, wenn man in obige Gleichung $i = 1$ einsetzt.

Damit kann der Ort i für das Bündelende aus den Shiftzahlen angegeben werden.

$$i = -452387 \cdot s_o - 72358 \cdot s_1 - 315238 \cdot s_2 - 330902 \cdot s_3 \bmod n_o$$

$$\left(n_o = 22 \cdot 89 \cdot 13 \cdot 23 = 585442 \right)$$

1.4.3.
Realisierter
Algorithmus

Bei der Realisierung werden im wesentlichen zwei "Tricks" verwendet. Statt der Multiplikation der Konstanten mit den entsprechenden Shiftzahlen werden die Konstanten rekursiv bei jedem Schritt zu einem Akkumulator addiert. Dabei werden die negativen Konstanten $-A_r$ als Komplement zur Periode M des Akkumulators dargestellt. Es ist gleichgültig, ob im Akkumulator um $-A_r$ (d.h. um A_r nach links im Bild) oder um $M-A_r$ (d.h. um $M-A_r$ nach rechts im Bild) fortgeschritten wird.
Außerdem tritt bei Addition von $M-A_r$ im Akkumulator ein Übertrag auf, solange die Akkumulation von $-A_r$ im darstellbaren Zahlenbereich $[-M+1,0]$ bleibt. Tritt bei Addition von $M-A_r$ kein Übertrag auf, dann würde der Zahlenbereich in negativer Richtung überschritten werden.

Der Akkumulator umfaßt fünf hexadezimale bzw. 20 binäre Stellen. Seine Periode ist somit

$$M = 16^5 = 2^{20} = 1048576$$

Im Akkumulator soll der Fehlerort FO aufgebaut werden. Als Fehlerort wird im Gegensatz zum Ort i des Bündelendes der Ort vom Bündelanfang unterstellt. Der Ort des Bündelanfangs ist bei $I = 11$ um $11 - 1 = 10$ größer als i. Außerdem wird der Fehlerort FO von hinten mit 1 beginnend gezählt. Deshalb wird der Akkumulator anfangs mit $10 + 1 = 11$ vorbelegt.

$$Acc := 11$$

Bei jedem der s_o Schiebeschritte von R_o wird die Konstante

$$A_o' = M - A_o = 1048576 - 452387 = 596189$$

zum Akkumulator addiert

$$Acc := Acc + A_o'$$

Dabei ist zu beachten, ob bei der Addition ein Übertrag im Akkumulator auftritt. Tritt keiner auf, dann wurde der Zahlenbereich des Akkumulators nach negativer Richtung hin überschritten. Diese Überschreitung wird durch Addition von n_o rückgängig gemacht. In der Rechnung modulo n_o (s. Formel für i) verändert eine Addition mit n_o das Ergebnis nicht.

Die übrigen Konstanten werden ebenfalls entsprechend umgeformt:

$$A_r' = M - A_r \qquad r = 1,2,3$$

Nachdem das Fehlerbündel in der unteren ("LOW") Hälfte vom Register R_o steht, werden die übrigen drei Register simultan solange verschoben, bis sie ebenfalls das Bündel enthalten. Jedes Register bleibt dann stehen, wenn es das Bündel enthält. Diese Tatsache zeigt die Maskenabfrage (CA . MB ⟶ CA) des Merkmalregisters MB an. Die verwendete resultierende Konstante K_s ist die Summe der jeweils noch beteiligten Konstanten A_r'.

Es bedeutet also (alle Additionen mod. M):

$K_1 = A_1' + A_2' + A_3' + n_o$	alle drei Register in Betrieb
$K_2 = A_2' + A_3' + n_o$	$R_2 + R_3$ in Betrieb, R_1 enthält Bündel
$K_3 = A_1' + A_3'$	$R_1 + R_3$ in Betrieb, R_2 enthält Bündel
$K_4 = A_1' + A_2'$	$R_1 + R_2$ in Betrieb, R_3 enthält Bündel
$K_5 = A_1'$	R_1 in Betrieb, $R_2 + R_3$ enthält Bündel
$K_6 = A_2'$	R_2 in Betrieb, $R_1 + R_3$ enthält Bündel
$K_7 = A_3'$	R_3 in Betrieb, $R_1 + R_2$ enthält Bündel

Dies sind die gespeicherten und bei Bedarf verwendeten Konstanten (s. Flußdiagramm). In K_1 und K_2 ist bereits jeweils n_o enthalten, da die entsprechenden Summen $A_1 + A_2 + A_3$ sowie $A_2 + A_3$ jeweils größer als n_o sind.

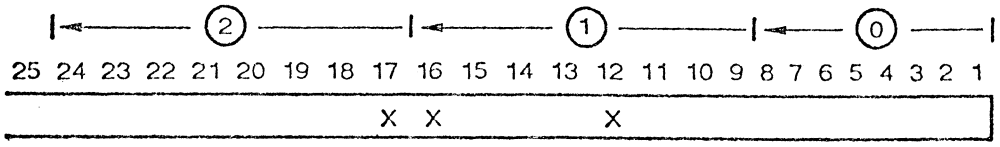
In der folgenden Tafel sind alle Konstanten und ihre Zusammenhänge nochmals dargestellt:

	Konstante hexadez.	Konstante dezimal	16^5-Konst. (dezimal)	Bemerkung
Vorbelegung	0000B	11		Bündellänge 1
Shiften R_o	918DD	596189	452387	A_o
kein Übertrag	8EEE2	585442		Polynomperiode n_o
$K_1(R_1,R_2,R_3)$	DF840	915520	133056	$(-n_o)$
$K_2(R_2,R_3)$	F12E6	987878	60698	$(-n_o)$
$K_3(R_1,R_3)$	9D8C4	645316	403260	
$K_4(R_1,R_2)$	A15F4	660980	387596	
$K_5(R_1)$	EE55A	976218	72358	A_1
$K_6(R_2)$	B309A	733338	315238	A_2
$K_7(R_3)$	AF36A	717674	330902	A_3

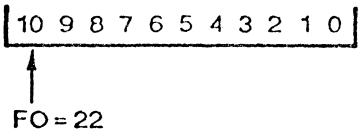
Wie die Nummern der fehlerhaften Bytes ermittelt werden und das Fehlermuster bytegerecht aufgeteilt wird, läßt sich am besten an einem Beispiel verfolgen.

Wie die Nummern der fehlerhaften Bytes ermittelt werden und das Fehlermuster bytegerecht aufgeteilt wird, läßt sich am besten an einem Beispiel verfolgen.

Beispiel:

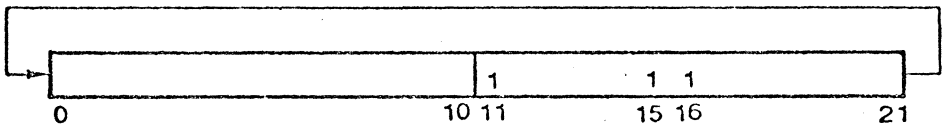


Codeblock mit Fehlern in den Stellen 12, 16, 17



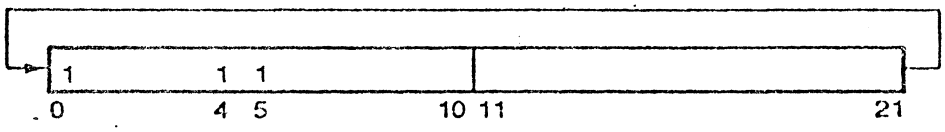
Fehlerbündel mit Nummern der Bündelstellen

Fehlerort



Register R_o
Inhalt nach Einlauf von $C'(x)$ bzw. $B(x)$
$B(x) \cdot x^{11}$

Nulltest erfüllt nach $s_o = 22 - 12 + 1 = 11$ Schiebetakten



Register R_o
Inhalt nach $s_o = 11$ Schiebetakten

Die anderen Register würden Übereinstimmung liefern nach s_r Takten:

$$s_1 = 89 - 12 + 1 = 78$$
$$s_2 = 13 - 12 + 1 = 2$$
$$s_3 = 23 - 12 + 1 = 12$$

Ort für Bündelende (von hinten mit 0 beginnend)

$$\begin{aligned} i &= -452387 \cdot 11 \\ &\quad - 72358 \cdot 78 \\ &\quad - 315238 \cdot 2 \\ &\quad - 330902 \cdot 12 \qquad \text{mod. } n_o \\ &= -15221481 \\ &= -26 \cdot 585442 + 11 \\ &= 11 \end{aligned}$$

Fehlerort FO (Bündelanfang von hinten mit 1 beginnend)

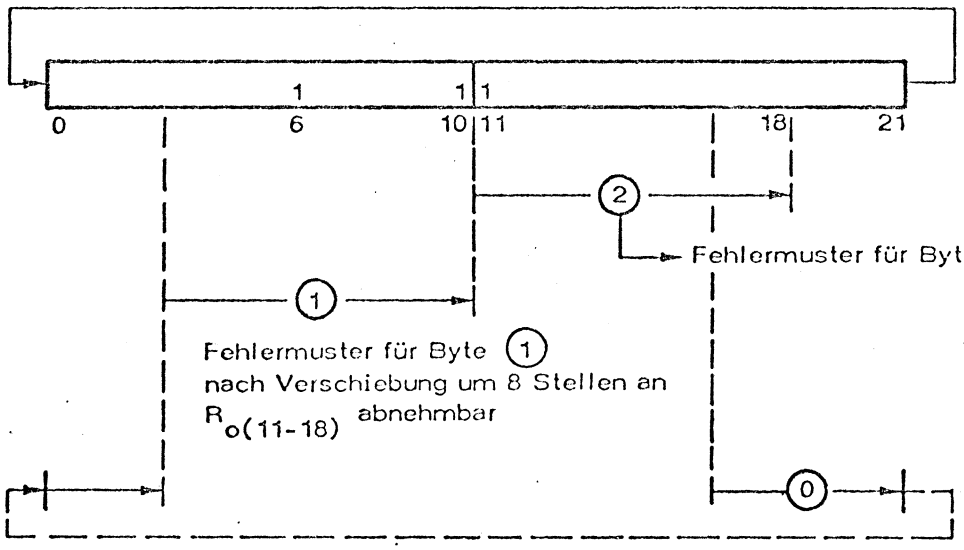
$$FO = i + 11 = 11 + 11 = 22$$
$$==$$

Fehlerbytes

22/8 = 2 Rest 6

- Rest 6 → Vorverschiebung
- 2 → Bytes (2), (1), (0)

werden mit einem bytegerechten Fehlermuster überlagert.



Register R_o
Inhalt nach 6 Vorverschiebungen

Fehlermuster für Byte (1) nach Verschiebung um weitere 8 Stellen an $R_{o(11-18)}$ abnehmbar
(die beiden untersten Stellen wegblenden)

COMPUTER GESELLSCHAFT KONSTANZ MBH

TECHNISCHER KUNDENDIENST

TECHNISCHER KUNDENDIENST

TELEFUNKEN COMPUTER

UNTERLAGENÄNDERUNG

Betrifft: TR 86 Unterlagen (verbal) Band 1 DFS/Reg 4 PE 30.1. Übertragungs-Steuerung	Änd. Index: 01 Datum: 27.2.74
Herausgeber: TC/VT 22 Schröder	Besteht aus 2 Blatt 1

Dok.-Nr. VT 460002-1201

Seite	Kartenart	Berichtigung
730	N-UE1	Signal e3 an St. 58 statt 53
850	N-UA1	Statt ta2s muß es $\overline{\text{ta2s}}$ heißen
850	N-UA1	Statt ta3s muß es $\overline{\text{ta3s}}$ heißen
610	N-US1	Signal am Ausgang G2/8 muß $\overline{\text{a2}}$ statt a2 heißen
680	N-US1	Signal am Ausgang F4/10 muß 114e statt 114c heißen
680	N-US1	Signal am Eingang F5/13 muß b3 statt $\overline{\text{b3}}$ heißen
860	N-UA1	Signal am Ausgang E6/12 muß $\overline{\text{faroe}}$ statt faroe heißen
970	N-DZ1	Signal am St. 43 muß $\overline{\text{normsp}}$ statt normsp heißen

SO/130 72/EZ23

Assembler-Beschreibung

Kapitel: PE 4.3.6

1. Auflage vom 10.7.76

umfaßt Blatt: PE 4.3.6-1 bis 4.3.6-480

INHALTSVERZEICHNIS

1. EINFÜHRUNG

Das Programm, dessen Anforderungen in diesem Kapitel beschrieben sind, führt den Namen

PS&WSP 430 ASS.

1.1. Ziele des Programms

Der größte Teil, der an die TR 440 angeschlossenen Peripherie ist nicht speziell für diese Anlage konzipiert worden. Daraus resultieren unterschiedliche Hardware-Schnittstellen, die mit sogenannten Anpaßwerken überbrückt werden. Diese Anpassung erfolgte bislang hardwareseitig, d.h., die Signale des Peripheriegerätes wurden durch logische Schaltungen umcodiert und dann an das E/A-Werk der TR 440 weitergeleitet und umgekehrt.

Für jedes neue Peripheriegerät war also ein neues Anpaßwerk zu entwickeln. Man hat sich deshalb entschlossen, ein programmierbares Anpaßwerk zu bauen, mit dem es nun möglich ist, unterschiedliche Peripheriegeräte an die TR 440 anzupassen.

Für dieses programmierbare Anpaßwerk ist eine Assemblersprache zu definieren und der zugehörige Assembler zu implementieren.

1.2. Beziehungen zu anderen Programmen

Das Programm (im folgenden kurz Assembler genannt) läuft unter dem TNS 440 an der TR 440-Anlage und wird mit dem Übersetzekommando gestartet. Der erzeugte Zielcode wird in der Datei "WSP 430 CODE" abgelegt (s. Kap. 4) und mit Hilfe des Programms MIKROCAS (s. Kap. 7) auf Magnetbandkassette ausgegeben.

1.3. Steuerung des Assemblers

Der Assembler übersetzt Programme, die in der speziell für das Anpaßwerk AW-WSP 430 entwickelten symbolischen Programmiersprache geschrieben wurden. Die Leistungen des Assemblers werden durch die Spezifikationswerte des Übersetze-Kommandos gesteuert. Folgende Spezifikationen werden ausgewertet: QUELLE, SPRACHE, NUMERIERUNG, MO, PROTOKOLL.

1.3.1. Spezifikation: QUELLE

1.3.1.1. Spezifikationswert: /Fremdstring

Im Normalfall wird die Quelle als Fremdstring angegeben werden, wenn sie als Kartenstapel vorliegt. Der Assembler erwartet die Quelle in diesem Fall in einem (vom Entschlüßler erstellten) Eingabegebiet. Die Daten des Eingabegebietes werden ihm im Startsatz übergeben.

```
Beispiel: QUE. = /
          C   BEGINN DER WSP430 ASS-QUELLE
```

1.3.1.2. Spezifikationswert: Datei

Wurde die Quelle vor dem Übersetzen durch die Kommandos des TN440 in eine Texthaltungskartei eingetragen (z.B. im Gespräch), kann der Dateiname als Spezifikationswert zu QUELLE angegeben werden. Liegt die Datei in der Standarddatenbasis &STDDB, braucht der Datenname nicht angegeben werden, da er vom Entschlüßler hinzugefügt wird. Liegen mehrere namensgleiche Dateien mit unterschiedlicher Generations-Versionsnummer in der gleichen Datenbasis, sollte deren Angabe nicht fehlen, weil sonst die Datei mit der größten Nummer genommen wird. Alle Angaben zur Datei werden dem Assembler vom Entschlüßler im Startsatz übergeben.

```
Beispiel: QUE. = &STDDB.  TEST (5.8)
          QUE. =          TEST (5.8)
          QUE. =          TEST
```

1.3.2.
Spezifikation: SPRACHE

Da fast alle Compiler und Assembler mit dem Übersetzekommando gestartet werden, soll auch der WSP 430-ASSEMBLER mit dem Kommando aktiviert werden. Da die Übernahme des Assemblers in die &OEFDB nicht genehmigt wurde, wird zu folgenden Notlösungen gegriffen: Der Assembler wird unter dem Namen PS&TASASSEMB durch das Kommando BINAEREIN aus einer privaten LF-Datei PS WSP 430 ASS in die &STDDB eingeschleust. Anschließend wird er mit dem Kommando UEBERSETZE, SPRACHE = TAS aktiviert.

1.3.3.
Spezifikation:
NUMERIERUNG

1.3.3.1.
Spezifikationswert: -STD-

Der Spezifikationswert -STD- wird vom Assembler folgendermaßen interpretiert: Liegt die Quelle als Fremdstring vor, werden die Protokollzeilen in Zehnerschritten durchnummeriert, beginnend bei 10. Steht die Quelle in einer Datei, werden die Satznummern der Quellzeilen unverändert für das Assemblerprotokoll übernommen.

1.3.3.2.
Spezifikationswert: (n,s)

Der Spezifikationswert (n,s) veranlaßt den Assembler, die Quellzeilen mit der Schrittweite s durchzunummerieren. Der Anfangswert ist n.

1.3.3.3.
Spezifikationswert:
H- (a,b)

Der Assembler blendet die letzten a-Spalten jeder Quellzeile aus und zwar unabhängig von deren Länge. Die Zeichenfolge ab dem b-ten Zeichen dieses Ausschnittes, wird vom Assembler als Numerierung interpretiert. Dabei wird abgeprüft, ob die Numerierung aufsteigend ist. Die Zeichenanzahl für die Numerierung ist auf maximal 6 festgelegt. Der Rest des Ausschnittes enthält im Normalfall ein Kurzzeichen für den Programmnamen.

Beispiel: NUM. = H- (8,4)
Zeilenlänge = 20 Spalten
QUELL-Zeile: C␣KOMMENTAR␣MODO 1230
PROTOKOLL-ZEILE OHNE O-Code
:␣␣1230␣␣␣C␣KOMMENTAR␣

1.3.4.
Spezifikation: MO

1.3.4.1.
Spezifikationswert:-

Die Angabe MO=- bedeutet: Der Assembler soll keinen Objektcode ausgeben. Es soll nur eine Syntax-Prüfung erfolgen. Der Assembler erstellt zwar intern den Objektcode, gibt ihn aber nicht aus. Die Codeausgabedatei WSP 430 CODE wird nicht kreiert.

1.3.4.2.
Spezifikationswert: -STD-

Die Angabe MO=-STD- verlangt vom Assembler die Ausgabe des Objektcodes nach vorgeschriebener Norm (siehe Kap. 3.4) in die Datei WSP 430 CODE.

1.3.4.3.
Spezifikationswert:
MO-NAME

Der Assembler kreiert mit dem MO-Namen eine Datei und gibt den Objektcode dorthin aus. Darüber hinaus gibt der Assembler den maximal 12 Zeichen langen MO-Namen als Assemblervorspann in die Datei WSP 430 CODE aus.

Beispiel: MO = WSP 430TEST

1.3.5.
Spezifikation: PROTOKOLL

1.3.5.1.
Spezifikationswert:-

Durch die Angabe PROT.=- wird die Protokollausgabe des Assemblers unterdrückt. Pseudobefehle zur Protokollgestaltung werden nicht interpretiert.

Ausnahme:

Sämtliche fehlerhaften Informationseinheiten werden trotzdem entweder (beim Abschnittsbetrieb) über den Schnelldrucker oder (beim Gesprächsbetrieb) über die Konsolen und bei eingeschaltetem Druckerprotokoll zusätzlich über den Schnelldrucker ausgegeben.

1.3.5.2.
Spezifikationswert: -STD-

PROT.=-STD- fordert vom Assembler ein Standardprotokoll ohne Objekt-Code (zum Protokoll-Format siehe Kap. 3.4). Das Standardprotokoll wird einzeilig ausgegeben.

1.3.5.3.
Spezifikationswert: O

Der Assembler druckt zusätzlich zum Standardprotokoll den erzeugten Objekt-Code aus. Dadurch wird das Format selbstverständlich geändert (zum Format siehe Kap. 3.4).

1.3.5.4.
Spezifikationswert: Z

PROT.=Z erwirkt vom Assembler eine zweizeilige Ausgabe des Standardprotokolls, d.h.: hinter jede Quellzeile wird eine Leerzeile ausgegeben.

1.3.5.5.
Spezifikationswert: R

PROT.=R impliziert PROT.=-STD-. Zusätzlich zum Standardprotokoll wird eine Crossreferenzliste ausgegeben (siehe Kap. 3.5.4).

1.3.5.6.
Spezifikationswert: &Z

PROT.=&Z impliziert den Spezifikationswert -STD-. Zusätzlich wird vom Assembler eine Speicherbelegungsliste ausgegeben (siehe Kap. 3.5.5), auch bei MO=-.

1.3.5.7.
Kombination von Spezifikationswerten

Die Spezifikationswerte Z, R, O, &Z sind beliebig kombinierbar, die Werte - und -STD- dürfen nicht zusammen mit anderen Spezifikationswerten auftreten.

Beispiel: ␣ *PRPTOKOLL (UEBERSETZE)=O'R'Z'&Z

Diese Vorbesetzung der Protokollspezifikation bewirkt die Ausgabe eines zweizeiligen Protokolls mit Objektcode-Ausgabe und dem Ausdruck der Crossreferenzliste und der Speicherbelegungsliste.

2. DEFINITION DES SPRACHUMFANGS

2.1. Grundelemente der Assemblersprache WSP 430 ASS

2.1.1. Zeichenvorrat

```
⟨Ziffer⟩          := Ø/1/2/3/4/5/6/7/8/9
⟨Buchstabe⟩       := A/B/C/D/E/F/G/H/I/J/K/L/M/
                     N/O/P/Q/R/S/T/U/V/W/X/Y/Z
⟨Sonderzeichen⟩:= ␣/'/.
```

Die in WSP 430 ASS zugelassenen Zeichen sind in allen TR 440-Eingabecodes vorhanden. Sämtliche Bausteine der Assemblersprache sind durch diese Zeichen darstellbar. Andere Zeichen dürfen nur in Kommentaren benutzt werden.

2.1.2. Bausteine der Assemblersprache

2.1.2.1. Namen

```
⟨Name⟩:= ⟨Buchstabe⟩ [⟨BUCHSTABE⟩ | ⟨ZIFFER⟩]Ø-15
```

Namen dienen zur Identifizierung von Befehlen, Pseudobefehlen und Registern. Sie dürfen maximal 6 Zeichen lang sein (siehe auch 2.5. Formatvorschriften). Für die zweite Ausbaustufe (noch nicht realisiert) des Assemblers sind lokale und globale Namen zur Darstellung von symbolischen Adressen vorgesehen. Globale Namen unterscheiden sich von den lokalen durch einen unmittelbar nach dem Namen folgenden Punkt.

2.1.2.2.
Konstanten
(allgemein)

```
⟨Konstante⟩   := ⟨Dezimalzahl⟩ | ⟨Oktalzahl⟩
⟨Dezimalzahl⟩:= ⟨Ziffer⟩ [⟨Ziffer⟩]o-n
⟨Oktalzahl⟩   := '⟨Oktalziffer⟩ [⟨Oktalziffer⟩]o-n'
⟨Oktalziffer⟩:= Ø/1/2/3/4/5/6/7
```

In WSP 430 ASS dienen die Konstanten zur expliziten Angabe der Ablageadresse einer Informationseinheit oder sie repräsentieren Teile einer IE. Durch die Größe des Bitfeldes innerhalb der IE wird der maximale Wert dieser Konstanten eindeutig festgelegt. Intern wird der Konstantenwert grundsätzlich rechtsbündig ins zugehörige Bitfeld eingetragen. Die Größe der Ablageadresse wird durch die Speicherkapazität des WSP 430-Anpaßwerkes bestimmt (max. 4095).

2.2. Informationseinheit (IE)

2.2.1. Aufbau der IE

```
⟨IE⟩ := [⟨Ablageadresse⟩]
        ⟨OT⟩ ⟨AT⟩ ⟨BT⟩ ⟨DT⟩ ⟨KO⟩ ⟨MSL⟩ ⟨NBA⟩ ⟨BSH⟩ ⟨BSN⟩
        [⟨Kommentar⟩] [⟨Kartenkennung⟩] |
        Cu ⟨Kommentar⟩ (siehe 2.4.) |
        ⟨Pseudobefehl⟩ (siehe 2.3.)
⟨Ablageadresse⟩:= ⟨Konstante⟩
⟨OT⟩ := reservierte Namen für: arithmetisch-logische Operationen
        (siehe 2.2.1.3.1.) | Spezial-Operationen (siehe 2.2.1.3.2.)
⟨AT⟩ := reservierte Namen für den A-Operanden
⟨BT⟩ := reservierte Namen für den B-Operanden|Ø|1|2|3
⟨DT⟩ := reservierte Namen für die Zielregister
⟨KO⟩ := ⟨Konstante⟩
⟨MSL⟩:= reservierte Namen für Merkerbefehle|Ø
⟨NBA⟩:= ⟨Konstante⟩
⟨BSH⟩:= reservierte Namen für Sprung-Bedingungen
⟨BSN⟩:= reservierte Namen für Sprung-Bedingungen
```

Außer Kommentaren und Pseudobefehlen belegt jede fehlerfreie Informationseinheit intern eine Speicherzelle des AW-WSP 430 à 48 Bits (MIBAP 1..48). Die Zeichnungen in 2.2.1.1. veranschaulichen den Aufbau einer Speicherzelle (= Mikrobefehl) des WSP 430-Anpaßwerkes. Die Lage eines Mikrobefehls im Mikroprogrammspeicher ist bestimmt durch die Ablageadresse (siehe 2.2.1.2.). Die Speicherkapazität des Mikroprogrammspeichers beträgt max. 4096 Mikrobefehle. Beim AW-WSP 430 ist sie auf 2048 Mikrobefehle festgelegt worden.
Der formale Aufbau der IE wird in Kapitel 2.5. Formatvorschriften beschrieben.

48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	MIBAP
1						6					5		2					5								8	1				4								7				4				4	1	Bitanzahl
PA	OT						AT					BT		DT					KO								P1	MSL				NB								BSH				BSN				P2	Feld-Name

Feld-Name	Wertebereich	Bedeutung
PA	0 - 1	Paritybit für die Ablage-Adresse (Ergänzung der gesetzten L-Bits auf gerade Anzahl)
OT	0 - 63	Operationsteil (ALU-Operation / Spezialoperation)
AT	0 - 31	A-Operanden-Teil
BT	0 - 3	B-Operanden-Teil
DT	0 - 31	Zielregister
KO	0 - 255	Konstanten-Teil
P1	0 - 1	Paritybit, ergänzt die gesetzten Bits MIBAP 22-48 auf gerade Anzahl
MSL	0 - 15	Merker setzen/löschen
NB	0 - 127	Adresse des Viererblocks für nächsten Befehl
BSH	0 - 15	Merker-Abfrage für bedingten oder unbedingten Sprung um 2
BSN	0 - 15	Merker-Abfrage für bedingten oder unbedingten Sprung um 1
P2	0 - 1	Paritybit, ergänzt die gesetzten Bits MIBAP 2-20 auf gerade Anzahl

Alle Paritybits werden automatisch gebildet.

2.2.1.2.
Ablageadresse

Der Zugriff auf einen Mikrobefehl erfolgt durch dessen Ablageadresse. Diese wird vor die zugehörige IE geschrieben (siehe 2.2.1. und 2.5.). In jedem Mikrobefehl wird im NB-, BSH- und BSN-Feld (siehe 2.2.1.7.) die Adresse des Folgebefehls angegeben, bei dem das Programm fortgesetzt wird. Es besteht folglich kein zwingender Grund, die Befehle einen nach dem anderen im Speicher abzulegen. Im Gegenteil, oft ist es gar nicht möglich, Befehle, die nacheinander durchlaufen werden, auch hintereinander abzulegen. Wenn der Programmierer gezwungen würde, mehrere Programmzweige gleichzeitig zu bearbeiten, könnte er sehr leicht den Überblick verlieren. Aus diesem Grund kann er die Ablageadresse für die IE'n explizit angeben. Fehlt diese Angabe, legt der Assembler die IE unmittelbar hinter den zuletzt bearbeiteten Mikrobefehl ab.

2.2.1.3.
Operationsteil (OT)

Für alle Operationen im OT-Feld wurden symbolische Namen festgelegt.

2.2.1.3.1.
ALU-Operationen

Hier wird die logische/arithmetische Funktion angegeben, nach welcher der A-Operand mit dem B-Operand miteinander zu verknüpfen/zu verrechnen sind. Das Ergebnis wird im Zielregister abgespeichert (siehe hierzu Kapitel 2.2.1.4.).

Bei allen Operationen + MC ist zu beachten, daß bei gesetztem Merker C (MC) auf das Rechenergebnis + 1 aufaddiert wird. Hierbei ist es gleichgültig, ob MC bereits zu Beginn des Befehls gesetzt war oder MC erst während des Rechenvorgangs selbst bei Übertrag gesetzt wurde (MC-Beschreibung siehe 2.2.1.6.). In beiden Fällen erfolgt die Verrechnung von MC noch im gleichen Befehl.

Zur Veranschaulichung der MC-Verrechnung nachstehende Tabelle mit allen vier möglichen Fällen:

Fall-Nr.	MC ist zu Beginn des Befehls:	Während der Rechenoperation entsteht Übertrag:	Auf das Rechenergebnis wird addiert:
1	gelöscht	nein	+0
2	gelöscht	ja	+1
3	gesetzt	nein	+1
4	gesetzt	ja	+1

In der folgenden Tabelle sind alle ALU-Funktionen aufgeführt und erläutert.

Symb. Name	Code intern	Funktion	Bemerkung
A	37	⟨DT⟩:=⟨AT⟩ *)	A-Operand
NA	2Ø	⟨DT⟩:= $\overline{\langle AT \rangle}$ *)	
AP	4Ø	⟨DT⟩:=⟨AT⟩+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf (>255)	
AMIN	17	⟨DT⟩:=⟨AT⟩-1 ⟨UTM⟩:=1 bei positivem | Ø bei negativem Ergebnis	
SHL	14	⟨DT⟩:=⟨AT⟩+⟨AT⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	
SHLP	54	⟨DT⟩:=⟨AT⟩+⟨AT⟩+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	
B	32	⟨DT⟩:=⟨BT⟩ *)	B-Operand
NB	25	⟨DT⟩:= $\overline{\langle BT \rangle}$ *)	
ZP	34	⟨DT⟩:=255 *)	
ZMIN	3	⟨DT⟩:=-1 *)	
ZN	43	⟨DT⟩:=-1+⟨MC⟩ ⟨UTM⟩:=1 bei positivem | Ø bei negativem Ergebnis	
ET	33	⟨DT⟩:=⟨AT⟩·⟨BT⟩ *)	Konjunktion A, B
NET	24	⟨DT⟩:= $\overline{\langle AT \rangle \cdot \langle BT \rangle}$ *)	
X	27	⟨DT⟩:=⟨AT⟩· $\overline{\langle BT \rangle}$ *)	
Y	22	⟨DT⟩:= $\overline{\langle AT \rangle}$·⟨BT⟩ *)	
V	3Ø	⟨DT⟩:= $\overline{\langle AT \rangle}$V(BT⟩ *)	
W	35	⟨DT⟩:=⟨AT⟩V $\overline{\langle BT \rangle}$ *)	
ETMIN	13	⟨DT⟩:=⟨AT⟩·⟨BT⟩-1 ⟨UTM⟩:=1 bei positivem | Ø bei negativem Ergebnis	
XMIN	7	⟨DT⟩:=⟨AT⟩· $\overline{\langle BT \rangle}$-1 ⟨UTM⟩:=1 bei positivem | Ø bei negativem Ergebnis	
WP	42	⟨DT⟩:=⟨AB⟩V $\overline{\langle BT \rangle}$+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf (>255)	

≢ Exklusiv Oder
V Disjunktion
· Konjunktion
+ Addition
− Subtraktion

Negierung wird durch Überstreichen dargestellt

*) ⟨UTM⟩:=Ø

Symb. Name	Code intern	Funktion	Bemerkung
VEL	36	⟨DT⟩:=⟨AT⟩V⟨BT⟩ *)	Disjunktion A, B
NVEL	21	⟨DT⟩:= $\overline{\langle AT\rangle V\langle BT\rangle}$ *)	
VELP	41	⟨DT⟩:=(⟨AT⟩V⟨BT⟩)+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	
AUT	26	⟨DT⟩:=⟨AT⟩≠⟨BT⟩ *)	Exclusiv- Oder A, B
NAUT	31.	⟨DT⟩:= $\overline{\langle AT\rangle \neq \langle BT\rangle}$ *)	
ADD	11	⟨DT⟩:=⟨AT⟩+⟨BT⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	Addition und Subtraktion A, B
ADDP	51	⟨DT⟩:=⟨AT⟩+⟨BT⟩+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	
SB	46	⟨DT⟩:=⟨AT⟩-⟨BT⟩-1+⟨MC⟩ ⟨UTM⟩:=Ø bei negativem | 1 bei positivem Ergebnis	
SBMIN	6	⟨DT⟩:=⟨AT⟩-⟨BT⟩-1 ⟨UTM⟩:=Ø bei negativem | 1 bei positivem Ergebnis	
ADET	1Ø	⟨DT⟩:=⟨AT⟩+(⟨AT⟩·⟨BT⟩) ⟨UTM⟩:=Ø| 1 bei Überlauf	Addition A plus f (A, B)
ADETP	5Ø	⟨DT⟩:=⟨AT⟩+(⟨AT⟩·⟨BT⟩)+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	
ADX	4	⟨DT⟩:=⟨AT⟩+(⟨AT⟩·$\overline{\langle BT\rangle}$) ⟨UTM⟩:=Ø| 1 bei Überlauf	
ADXP	44	⟨DT⟩:=⟨AT⟩+(⟨AT⟩· $\overline{\langle BT\rangle}$)+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	
ADVL	15	⟨DT⟩:=⟨AT⟩+(⟨AT⟩V⟨BT⟩) ⟨UTM⟩:=Ø| 1 bei Überlauf	
ADVLP	55	⟨DT⟩:=⟨AT⟩+(⟨AT⟩V⟨BT⟩)+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	
ADW	16	⟨DT⟩:=⟨AT⟩+(⟨AT⟩V $\overline{\langle BT\rangle}$) ⟨UTM⟩:=Ø| 1 bei Überlauf	
ADWP	56	⟨DT⟩:= ⟨AT⟩+(⟨AT⟩V $\overline{\langle BT\rangle}$)+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	

*) ⟨UTM⟩:=Ø

Symb. Name	Code intern	Funktion	Bemerkung
ETADW	12	⟨DT⟩:=(⟨AT⟩·⟨BT⟩)+(⟨AT⟩V $\overline{⟨BT⟩}$) ⟨UTM⟩:=Ø| 1 bei Überlauf	Addition f (A, B) plus f (A, B)
VLADX	5	⟨DT⟩:=(⟨AT⟩· $\overline{⟨BT⟩}$)+(⟨AT⟩V⟨BT⟩) ⟨UTM⟩:=Ø| 1 bei Überlauf	
ETADWP	52	⟨DT⟩:=(⟨AT⟩·⟨BT⟩)+(⟨AT⟩V $\overline{⟨BT⟩}$)+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	
VLADXP	45	⟨DT⟩:=(⟨AT⟩· $\overline{⟨BT⟩}$)+(⟨AT⟩V⟨BT⟩)+⟨MC⟩ ⟨UTM⟩:=Ø| 1 bei Überlauf	

2.2.1.3.2.
Spezial-Operationen

. Hier werden spezielle Operationen angegeben, die nicht über die ALU laufen. Die ALU wird zwar mit angesteuert, aber nicht ausgewertet. Der Null-Merker (NM) und der Übertragsmerker (UTM) können sich ändern.

2.2.1.3.2.1.
Spezial-Operationen
für Datenspeicher

Im Datenspeicher (DSP) können die Registerinhalte (je acht Bit und Parity) des AT-Feldes (siehe 2.2.1.4.) abgelegt und bei Bedarf wieder in die Register des DT-Feldes gebracht werden. Die Kapazität beträgt 256 Registerinhalte (bei Hardware-Änderung erweiterbar bis 1024). Der Zugriff zum DSP erfolgt über die Transportbefehle DBSKO, DBLKO, DBSHN, DBLHN, wobei die Adressierung des DSP einmal über KO (feste Adresse) oder über HN-Register (variable Adresse) erfolgt. Hierzu nachstehende Erläuterungen.

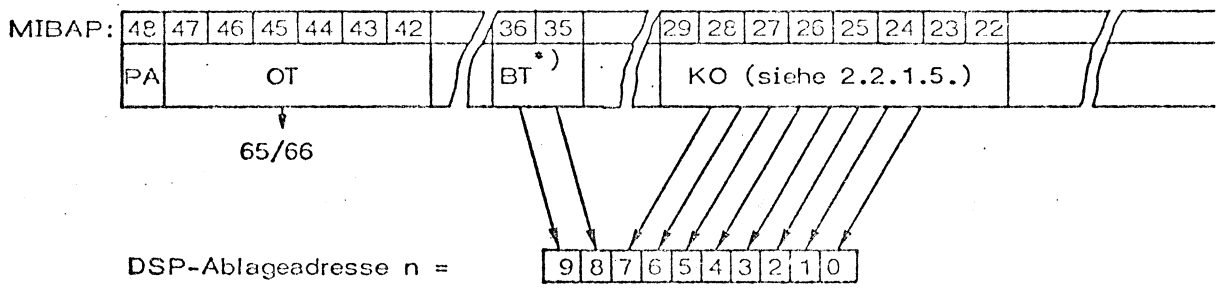
Symb. Name	Code intern	Funktion	Bemerkung
DBSKO	65	n:=BT.256+⟨KO⟩ ⟨n⟩:=⟨AT⟩	Transport-Befehle für Datenspeicher n=DSP-Ablageadresse Ø≤n≤1Ø23 BT≙Adreß-Zahlenwert
DBLKO	66	n:=BT.256+⟨KO⟩ ⟨DT⟩:=⟨n⟩	
DBSHN	75	n:=BT.256+⟨HN⟩ ⟨n⟩:=⟨AT⟩	
DBLHN	76	n:=BT.256+⟨HN⟩ ⟨DT⟩:=⟨n⟩	

Es bedeutet:

a) DBSKO: Bringe AT-Registerinhalt (1 Byte) nach DSP.
DBLKO: Bringe abgespeichertes Byte nach DT-Register.

Hierbei wird die Ablageadresse n im DSP durch BT und KO wie folgt gebildet:

Mikrobefehl:



Beispiel: '0000'␣DBSKO␣␣DR␣␣1␣LE␣␣␣␣100␣ ..... etc.

ergibt: n = 1·256+100=356

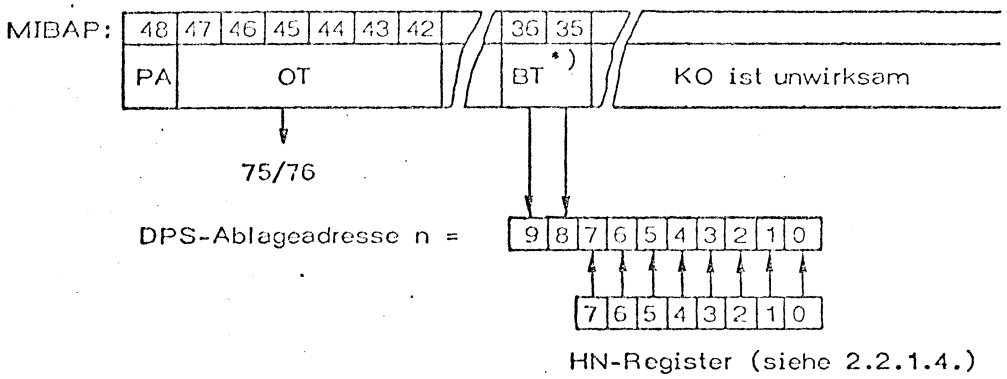
Der Inhalt des Registers DR wird in die DSP-Zelle mit der Adresse 356 geschrieben.

*) siehe 2.2.1.4.

b) DBSHN: Bringe AT-Registerinhalt (1Byte) nach DSP.
DBLHN: Bringe abgespeichertes Byte nach DT-Register.

Hierbei wird die Ablageadresse n im DSP durch BT und HN-Register-Inhalt wie folgt gebildet:

Mikrobefehl:



Beispiel: '0000' ␣ DBSHN␣␣ OP␣␣ 0 ␣ LE ␣␣␣␣␣␣ 0 ␣ ..... etc.

Im HN-Register steht 120. Somit ergibt sich n = 0·256+120=120

Der Inhalt des Registers OP wird in die DSP-Zelle mit der Adresse 120 geschrieben.

ACHTUNG!

Da die derzeitige DSP-Kapazität nur 256 Byte beträgt, müssen Bit 8 und 9 der DSP-Ablageadresse stets "0" sein, d.h., im BT-Feld muß bei allen DSP-Operationen immer "0" oder "LE" stehen.

2.2.1.3.2.2.
Spezial-Operationen für WSP-ECC-Logik

Dieses sind spezielle Befehle für WSP 430-Anpaßelektronik. Beschreibung siehe unter WSP 430-Schnittstellenbeschreibung. Für andere Schnittstellen stehen diese Operationen frei zur Verfügung, wobei der symb. Name bleibt.

Symb. Name	Code intern	Funktion	Bemerkung
NFECC	60	normieren bei ECC-Fehler	Spezial-Operations-teile
LFB	61	laden der Fehlerbedingungen nach SC-Register	
SHP0	62	Shiften Polynomgenerator P0	
SHP123	63	Shiften Polynomgenerator P1, P2, P3	

*) siehe 2.2.1.4.

2.2.1.4.
Operandenteile (AT, BT) und Zielregister (DT)

Als Operandenteile gelten:

a) Alle Register, die als A- und/oder B-Operand für die ALU (siehe 2.2.1.3.1.) zu verwenden sind. Der Inhalt der AT-Register kann in den Datenspeicher abgelegt werden (siehe 2.2.1.3.2.1.).

b) Spezielle Steuerbefehle wie Programm-Stop und Datenspeicher-Seitenadressierung.

c) Der Konstantenteil KO (siehe 2.2.1.5.) als B-Operand.

Als Zielregister gelten alle Register des DT-Feldes. Sie können vom Datenspeicher geladen werden.

Die Operandenteile und Zielregister haben folgende Namen:

Code intern	0	1	2	3	4	5	6	7	10	11	12	13	14
AT	LE	AB	LG	FR	DR	DW	PE	IS	MA	OP	HC	HD	KA
DT	LA	AB	LG	FR	DR	DW	PE	IS	MA	OP	HC	HD	KR
BT	LE	AB	KO	DR									
	0	1	2	3	*)								

Code intern	15	16	17	20	21	22	23	24	25	26	27	30	31
AT	FC	FS	MS	HE	PL	HF	HG	HI	HK	FT	—	HM	HN
DT	LE	FS	MS	HE	PL	HF	HG	HI	HK	Z1	Z2	HM	HN

Code intern	32	33	34	35	36	37
AT	HO	HP	MB	ME	SC	SP
DT	HO	HP	FC	FT	ST	—

*) Zahlen nur bei den Operationen DBSKO, DBLKO, DBSHN, DBLHN zur Seitenadressierung des Datenspeichers (Bit 8,9) (siehe 2.2.1.3.2.1.)

Alle Register sind 8 Bit lang und haben ein Paritybit.

Beschreibung der Operandenteile und Zielregister mit WSP 430-Schnittstelle

nur WAW 431

Verwendbar als A-Operand (AT-Feld) A BUS	Verwendbar als B-Operand (BT-Feld) B BUS	Verwendbar als Zielreg. (DT-Feld) D BUS	Verwendet als Schnittstellen-Register für Rechner	Verwendet als Schnittstellen-Register für WSP*	Spezielle Eigenschaften, Bemerkungen
AB	AB	AB			
DR	DR	DR		DR 1)	1) Daten bei Lesen (DR→DW) und Schreiben (KA→DR(DW))
DW		DW	DW 2)		2) Datensender. Lesen/Schreiben (siehe DR)
FR		FR			
FC		FC		FC 3)	3) BUSOUT-Register
FS		FS		FS 4)	4) BUSIN-, Gerätefehler- u. Modul-Adreß-Register (→FT)
FT		FT		FT 5)	5) TAGBUS-Reg. (Bit 0-4), Selektierung der FS-Eingabe (Bit 6,7)
HC		HC			
HD		HD			
HE		HE			
HF		HF			
HG		HG			
HI		HI			
HK		HK			
HM		HM			
HN		HN			DSP-Adreß-Register (von MIP anwählbar)
HO		HO			
HP		HP			
IS		IS			
KA			KA 6)		6) Datenempfänger, Durchschaltbedingung: STP
		KR	KR 7)		7) Quittierung: STP, AF, EMP, GA, FE, IE, AN2, AN1
	KO				Konstante (MIBAP 22-29). Verwendbar auch als DSP-Adresse
LE	LE	LE			Leer = Pseudoregister mit Dauerinhalt "0"
		LA			Register für Lampenanzeige
LG		LG			
MA		MA			
MB				MB	ECC-Fehlererkennung 1 bei Lesen
ME				ME	ECC-Fehlermuster
MS		MS			
OP		OP			Register bitweise abfragbar über BSH/BSN-Feld
PE		PE			
PL		PL			
SC				SC	ECC-Fehlererkennung 2 bei Lesen/Schreiben und Übertragungsfehler
		ST		ST	Steuermeldungen für Lesen/Schreiben und ECC-Logik
SP					Programmierbarer Stop (kein Reg.)
		Z1		Z1	Datenlängenzähler $2^0 - 2^7$
		Z2		Z2	Datenlängenzähler $2^8 - 2^{15}$

Verwendbar für ALU als A-Operand (AT-Feld) ABUS	B-Operand (DT-Feld) BBUS	Zielreg. (DT-Feld) DBUS	Verwendbar als Schnittstellen-Senderegister	Schnittstellen-Empfangsreg.	Spezielle Eigenschaften, Bemerkungen
AB	AB	AB	AB[1]		[1] Nur bei Magazin-Nachverdrahtung (9 Ltg.)
DR	DR	DR	DR	DR	Außerdem Zwischenpuffer für KA-Reg. (von HW steuerbar)
DW		DW	DW	DW	
FR		FR			
FC		FC	FC	FC[2]	[2] Nur bei Magazin-Umverdrahtung und zusätzl. MX-Logik
FS		FS[3]		FS	[3] Nur bei zusätzl. MX-Logik i.d.Schnittst.-Elektronik oder bei Wegfall als Empf.-Register nur bei zusätzl. Verdrahtung
FT		FT			Kein Register; dieser Name steht frei zur Verfügung
HC		HC			
HD		HD			
HE		HE	HE		
HF		HF			
HG		HG			
HI		HI	HI	HI	
HK		HK	HK	HK	
HM		HM	HM[1]		[1] Nur bei Magazin-Nachverdrahtung 9 Ltg.)
HN		HN	HN[1]		Außerdem DSP-Adreß-Register (v.Progr. steuerbar), [1] wie bei HM
HO		HO			
HP		HP			
IS		IS		IS[4]	[4] Nur bei Magazin-Umverdrahtung (MX-Logik vorh.)
KA		KA[3]	KA[1]	KA	[1] wie bei HM [3] wie bei FS, programmierbar über DT 29
	KO				Progr. Konstante, außerdem DSP-Adreß-Register (vom Programm steuerbar).
		KR			Kein Register; dieser Name steht frei zur Verfügung.
		LA			Register für Lampenanzeige.
LE	LE	LE			Pseudoregister, dessen Inhalt ständig "0" ist (Leer)
LG		LG			
MA		MA	MA		
MB		MB[3]	MB	MB	[3] wie bei FS, programmierbar, z.B. über DT 12 (KR)
ME		ME[3]	ME	ME	[3] wie bei FS, programmierbar, z.B. über DT 22 (Z1)
MS		MS		MS[2]	[2] wie bei FC
OP		OP	OP[1]		[1] wie bei HM, per Programm bitweise abfragbar (BSH, BSN)
PE		PE		PE	
PL		PL	PL[1]		[1] wie bei HM
SC		SC[3]		SC	[3] wie bei FS
SP					Programmierbarer Stop (Kein Register)
		ST			Kein Register; dieser Name steht frei zur Verfügung
	0				DSP-Adreßbereich 0-255 (= Bit 8,9 = 0).
	1-3[5]				[5] Durch Umbau der R - WS 36 u. Magazin-Umverdrahtung Erweiterung des DSP bis 512 bzw. 1K möglich (BT 1-3)
		Z1			Kein Register; diese Namen stehen zur Verfügung
		Z2			

2.2.1.5.
Konstantenteil (KO)

Der Konstantenteil umfaßt die MIBAP-Bits 22-29 (KO0-7) des Mikrobefehls. Damit entspricht er in seiner Größe einem Register. Der Konstantenteil hat folgende Bedeutung:

a) Verwendung als B-Operand (siehe 2.2.1.4.), adressierbar durch (⟨BT⟩:=KO), wobei BSH ≠ 17 sein muß.

b) Verwendung zur Mikrobefehls-Seitenadressierung (MIBAR Bit 9-11), adressierbar durch BSH = 17 (siehe 2.2.1.7.3. und 2.2.1.7.4.)

c) Verwendung zur Datenspeicheradressierung (Bit 0-7), adressierbar durch DBSKO oder DBLKO (siehe 2.2.1.3.2.1.)

Der maximale Wert ist durch die Länge des Konstantenfeldes auf 255 oder '377' festgelegt. Der Wert ist als Dezimal- oder Oktalzahl anzugeben.

2.2.1.6.
Merker Setzen/Löschen
(MSL-Teil)

Der MSL-Teil enthält die Anweisung zum Setzen oder Löschen von acht Merkern:

Name	Code intern	Wirkungsweise
0	0	keine Veränderung eines Merkers
LMO	1	Merker MO := 0
SMO	2	Merker MO := L
LMI	3	Merker MI := 0
SMI	4	Merker MI := L, wenn Index anliegt
LMU	5	Merker MU := 0; L, nach 20 ms
DPL	6	DIAGNOSTIK PROGRAMM LADEN
LMC	7	Merker MC := 0
SMC	10	Merker MC := L
LMD	11	Merker MD := 0
LM5	12	Merker M5 := 0
SM5	13	Merker M5 := L
LM6	14	Merker M6 := 0
SM6	15	Merker M6 := L
LM7	16	Merker M7 := 0
SM7	17	Merker M7 := L

NM: Null-Merker wird automatisch auf "L" gesetzt, wenn das Ergebnis einer ALU-Operation gleich 0 ist. NM ist nicht adressierbar.

Die Merker MO, MI, MU, MC, MD, M5, M6 und M7 können im BSH- bzw. BSN-Teil (siehe 2.2.1.8.) des nächsten Mikrobefehls abgefragt werden und gehen somit in die Berechnung der Adresse des Folgebefehls ein. Sie sind für die Programmverzweigung relevant.

Beschreibung der Merker

Merker	Funktion im WAW 431	PAW 405
M0 M5 M6	Sie stehen dem Programmierer frei zur Verfügung und können nur durch das Mikroprogramm (MIP) verändert werden.	
M7	leitet die automatische Datenphase ein. Er muß vom MIP gesetzt werden. Das Löschen erfolgt automatisch bei Ende der Datenphase oder durch das MIP.	ist frei verfügbar. Setzen nur vom MIP möglich, Löschen vom MIP oder hardwaremäßig von der Schnittstelle aus möglich.
MI	wird nur gesetzt, wenn vorher vom MIP SMI gegeben wurde (Setzen des Vormerkers) und dann	
	die Indexmeldung ansteht. Die Indexmeldung setzt dann MI direkt.	ein Setzsignal von der Schnittstellen-Hardware gegeben wird.
	Löschen nur vom MIP möglich.	
MU	kann nur vom MIP gelöscht werden und wird nach 25 ms wieder automatisch gesetzt.	
	Automatisches Setzen außerdem bei Übertragungsfehler.	Außerdem direktes Setzen von der Schnittstellen-Hardware möglich.
DPL	Diagnostik-Programm Laden = Übernahmetakt (kein Merker!) für die Diagnostik-Programm-Bytes in den MIP-Speicher. Siehe hierzu auch DPL-Flußdiagramm in Kap. 1.2.2.5. und Strukturdiagramm in Kap. 1.6. Mit jedem DPL wird ein Rechner-Byte à 8 Bit (sechs Daten-Bits und zwei Steuer-Bits) in einen Zwischenpuffer übernommen. Die beiden Steuer-Bits (Bit 6 + 7) sind für die Software bedeutungslos. Da ein MIP-Wort aus 48 Bits und 12 Adreßbits besteht, müssen pro MIP-Wort 10 Rechner-Bytes übernommen und somit 10 x DPL gegeben werden. Der erlaubte Abstand zwischen zwei aufeinanderfolgenden DPL beträgt 2,2 us - 130 µs. Nach jedem 10. DPL wird das fertig aufbereitete MIP-Wort in den MIP-Speicher geschrieben. Hierzu wird das MIP angehalten (Taktkette wird gestoppt). Nach Beendigung des Einschreibens wird das MIP automatisch fortgesetzt. Aus Hardware-Gründen muß nach dem 10. DPL-Befehl ein Leerbefehl folgen. In diesem Leerbefehl sind nur Abfragen erlaubt (z.B. MC). Hardware-Fehler bei DPL: Der Merker C (MC) muß zu Beginn des DPL-Betriebes vom MIP gesetzt werden. Tritt vor oder während des Schreibens ein Hardwarefehler auf, so wird MC automatisch gelöscht. Hierdurch erfolgt eine Fehlermeldung vom	

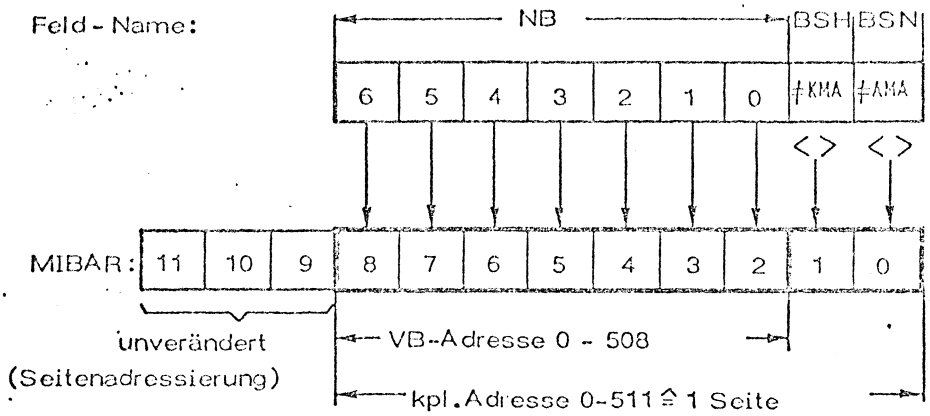
Merker	Funktion im WAW 431	PAW 405
DPL (Fortsetzg.)	MIP an den Rechner und der DPL-Betrieb wird abgebrochen. Folgende Hardwarefehler werden erkannt: a) Parityfehler der MIP-Wort-Adresse (FEING) b) MIP-Speicherfehler: Eingeschriebenes MIP-Wort falsch (FSPAPR).	
MC	Setzen: a) durch das MIP (<MSL>:=SMC) b) bei einer ALU-Operation mit Interncode 40-56, wenn Übertrag entsteht (<UTM>=L). Er dient hierbei als Übertrags-Zwischenspeicher und wird automatisch verrechnet. Löschen: a) durch das MIP (<MSL>:=LMC) b) automatisch bei DPL, wenn ein Hardwarefehler auftritt.	
MD	Setzen: automatisch bei Ende der Datenübertragung eines Bytes von und zum WSP. Löschen: a) durch das MIP (<MSL>:=LMD) mit Takt T5 b) automatisch bei Sync-Byte suchen.	Setzen: mit Takt T5, wenn vorher Zwischenmerker ZMD hardwaremäßig von der Schnittstelle aus gesetzt wurde. Löschen: a) durch das MIP (<MSL>:=LMD) mit Takt T5 b) hardwaremäßig von der Schnittstelle aus.

2.2.1.7.
Aufbereitung der Adresse des Folgebefehls (MIBAR)

Das WAW 431 bzw. PAW 405 hat keinen Befehlsfolgezähler wie z.B. der TR 440. In jedem Befehlszyklus muß deshalb die Adresse des Folgebefehls neu errechnet werden. Da max. 4K-Befehle zu adressieren sind, muß MIBAR 12 Bits (MIBAR 0-11) umfassen. Die Aufbereitung von MIBAR kann auf vier verschiedene Arten erfolgen, welche nachstehend beschrieben sind:

2.2.1.7.1.
Adressierung über NB-, BSH- und BSN-Teil *)

Hierfür muß BSH ≠ KMA und BSN ≠ AMA sein.
Dies ist die normale Adressierungsart. MIBAR setzt sich wie folgt zusammen:



Es können hiermit alle 512 Mikrobefehle innerhalb einer Seite (1 Seite ≙ 512 Mikrobefehle) direkt adressiert werden. Seitenwechsel ist nicht möglich.

Im NB-Feld wird die 1. Adresse eines Viererblocks (VB) explizit angegeben. Sie entspricht den Bits 2-8 von MIBAR und muß durch vier teilbar sein. Da ein Seitenwechsel nicht möglich ist, geht nur der seitenrelative Teil der Adresse in die Berechnung der Folgeadresse ein. Der höherwertige Teil der Adresse (Seitenadressierung) dient dem Assembler zur Kontrolle, ob die aktuelle Seite nicht verlassen wird.

Die BSH- und BSN-Felder sind je vier Bit lang und wählen die bedingten Sprunganweisungen aus. BSH bestimmt das Bit 1, BSN das Bit 0 von MIBAR. Somit werden BSH und BSN zur Bestimmung der Einzeladressen innerhalb des durch NB adressierten Viererblocks herangezogen.

Beispiel:

	VB-Adresse: (NB-Feld)	<BSH>	<BSN>	kpl. Adresse
1 VB	1240	0	0	1240
	1240	0	1	1241
	1240	1	0	1242
	1240	1	1	1243
nächste VB-Adr.:	1244	0	0	1244
	⋮	⋮	⋮	⋮
		usw.		

*) NB = Nächster Befehl
BSH= Bedingter Sprung hoch
BSN= Bedingter Sprung niedrig

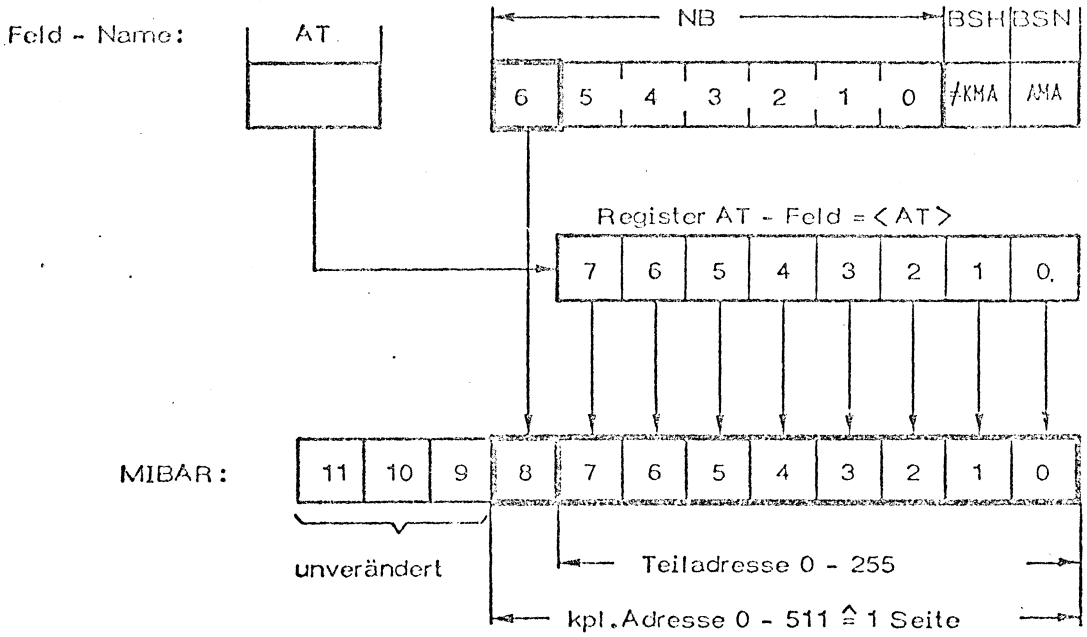
Symbolische Namen für BSH- und BSN-Teil

Code intern	Symb. Name BSH	BSN	Bemerkungen
0	SH0	SN0	Direkte Adressierung der Befehle eines Viererblocks (unbedingter Sprung). SH0, SN0 = 0; SH2, SN1 = L
1	SH2	SN1	
2	SM0	SMI	Abfragen der Merker M0-M7, MI, MC, MD
3	ILN	SMC	ILN: Ansprungbedingung für In-Line-Diagnostikprogramme. Es ist erfüllt, wenn Schalter SB auf INLINE steht; des weiteren muß Schalter STMA auf RUNDL oder bei LA-Reg. = 0 auf FEHLER stehen
4	SMD	SM5	
5	SM6	SM7	
6	OP0	OP1	Bitweise Abfragen des Registers OP
7	OP2	OP3	
10	OP4	OP5	
11	OP6	OP7	
12	UTM	SNM	Abfragen von Kanalmeldungen und der Signale des Rechenwerkes. SNM (Null-Merker) wird automatisch vom ALU gesetzt UTM wird automatisch gesetzt bei ALU-Übertrag FM1 wird gesetzt, wenn Fehler in der Kanal-Anpassung und im Gerät auftreten
13	STM	ZM1	
14	FM1	SPM	
15	SNM	BM1	
16	TFK	SMU	Abfrage des Merkers MU TFK: Merker für Rechner-Doppelzugriff: gesetzt: Rechner 1 gelöscht: Rechner 2
17	KMA	AMA	Änderung von MIBAR in den Bits 9-11 (KMA) bzw. in den Bits 0-7 (AMA)

Gesetzter Zustand (= L) bedeutet Sprung.

2.2.1.7.2.
Adressierung über Register-Inhalt des AT-Feldes und NB6

Hierfür muß BSH ≠ KMA und BSN = AMA sein.
MIBAR setzt sich wie folgt zusammen:

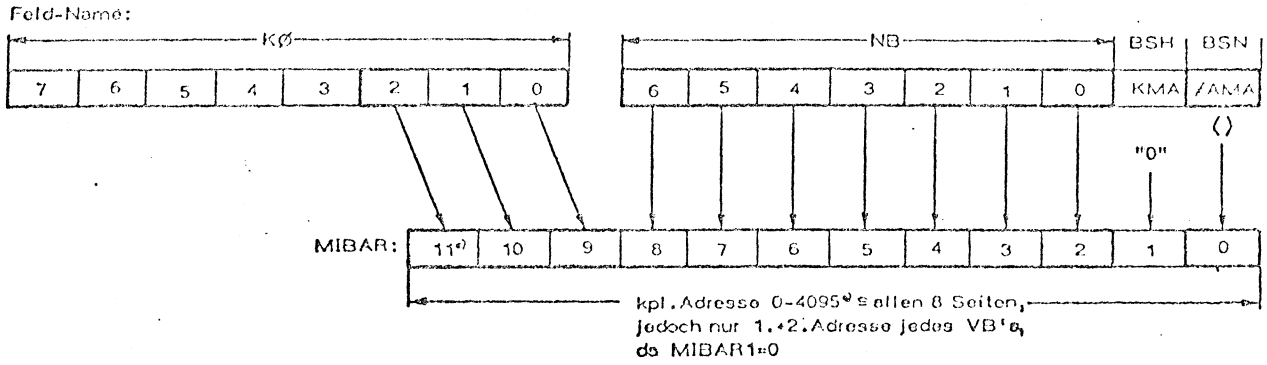


Es wird der durch AT adressierte Registerinhalt als Folgeadresse ausgewertet. Da jedes Register maximal den Wert 255 enthalten kann, kann auf diese Weise lediglich eine Adresse 0-255 angesteuert werden. Um alle Befehle innerhalb einer Seite adressieren zu können, wird Bit 8 von MIBAR durch das 6. Bit von NB (NB6) direkt auf "0" oder "L" gesetzt. Die Bits 9-11 von MIBAR (Seitenadressierung) werden nicht verändert, d.h., der Folgebefehl muß in der gleichen Seite liegen.

Im NB-Feld ist die erste Adresse der entsprechenden Halbseite anzugeben, z.B. wenn in die zweite Halbseite (256-511) gesprungen werden soll, muß in NB 256 angegeben werden.

2.2.1.7.3.
Adressierung über Konstantenteil (KO 0-2), NB und BSN

Hierfür muß BSH = KMA und BSN ≠ AMA sein.
MIBAR setzt sich wie folgt zusammen:



Die Bits KO0, KO1, KO2 bestimmen direkt die Bits 9, 10 und 11 von MIBAR, NB bestimmt direkt die Bits 2-8 vom MIBAR. MIBAR 1 wird auf jeden Fall auf "0" gesetzt, während MIBAR 0 je nach dem Ergebnis der Sprungbedingung in BSN der Wert "0" oder "L" zugewiesen wird.

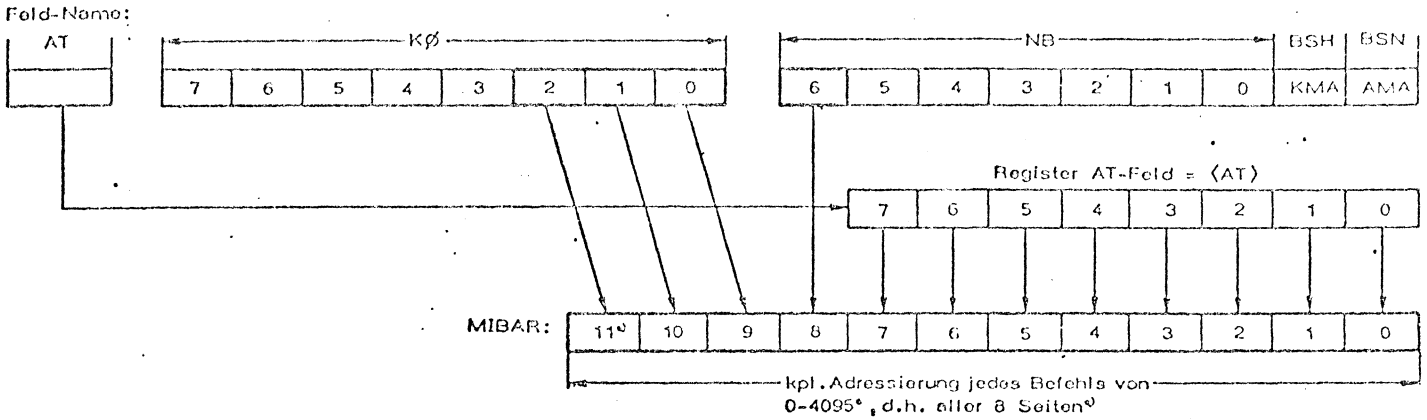
Auf diese Weise sind jeweils die ersten beiden Befehle jedes Viererblocks auf den Seiten 0-7*) adressierbar.

Im NB-Feld muß die absolute VB-Zieladresse angegeben werden. Der Assembler überprüft, ob diese Adresse mit der angegebenen Konstante (KO) angesprungen werden kann.

*) Da die derzeitige Speicherkapazität des MIP-Speichers nur 2K Befehle beträgt, muß MIBAR 11 stets "0" sein, d.h., es dürfen nur die ersten vier Seiten (Seite 0-3 = 0-2047) adressiert werden.

2.2.1.7.4.
Adressierung über Konstantenteil (K00-2), NB6 und Registerinhalt des AT-Feldes

Hierfür muß BSH = KMA und BSN = AMA sein.
MIBAR setzt sich wie folgt zusammen:



Die Bits 0-7 der Mikrobefehlsadresse (MIBAR) ergeben sich aus den acht Bits des durch AT adressierten Registers, die Bits 9-11 aus den Bits 0-2 des Konstantenteils. Bit 8 von MIBAR wird durch NB6 bestimmt. Damit sind jeweils alle Befehle der Seiten 0-7*) adressierbar.

Im NB-Feld ist die erste Adresse der entsprechenden Halbseite anzugeben.

*) Da die derzeitige Speicherkapazität des MIP-Speichers nur 2K Befehle beträgt, muß MIBAR 11 stets "0" sein, d.h., es dürfen nur die ersten vier Seiten (Seite 0-3 = 0-2047) adressiert werden.

Bereich der Folgeadresse

KO 2	KO 1	KO 0	NB 6	Bereich (dezimal)	Seite
0	0	0	0	0 - 255	0
0	0	0	L	256 - 511	0
0	0	L	0	512 - 767	1
0	0	L	L	768 - 1023	1
0	L	0	0	1024 - 1279	2
0	L	0	L	1280 - 1535	2
0	L	L	0	1536 - 1791	3
0	L	L	L	1792 - 2047	3
L	0	0	0	2048 - 2303	4
L	0	0	L	2304 - 2559	4
L	0	L	0	2560 - 2815	5
L	0	L	L	2516 - 3071	5
L	L	0	0	3072 - 3327	6
L	L	0	L	3328 - 3583	6
L	L	L	0	3584 - 3839	7
L	L	L	L	3846 - 4095	7

*)

*) im derzeitigen 2K-Ausbauzustand des MIP-Speichers nicht erlaubt.

2.3. Pseudobefehle

Pseudobefehle sind Anweisungen an den Assembler. Sie erzeugen keinen Ausgabecode.

Folgende Pseudobefehle werden implementiert:

Befehl	Wirkung
ENDE	zeigt dem Assembler das statische Ende einer Quelle an
PAUS	Protokollierung des Übersetzerprotokolls ausschalten
PEIN	Protokollierung des Übersetzerprotokolls einschalten
NS	Fortsetzung des Übersetzerprotokolls auf neuer Seite
NLn	Zeilenvorschub um n Zeilen, für $1 \leq n \leq 7$
GLCH	definiert einen Gleichsetzungsnamen
BASIS	Verschiebung sämtlicher folgenden Adressen
SBASIS	Verschiebung der Folgeadresse des nächsten Befehls

Die Pseudobefehle, die die Protokollgestaltung betreffen, werden nur ausgewertet, wenn überhaupt ein Übersetzerprotokoll erstellt wird. Sie selbst werden nicht protokolliert.

Des weiteren gibt es die Pseudobefehle GLCH, BASIS und SBASIS. Sie eröffnen die Möglichkeit, mehrere Programme (z.B. off-eine-Testprogramme) zu ketten, ohne daß die Ablageadressen der Befehle umgeschrieben werden müssen. Es ist jedoch nur eine Adreßverschiebung in positiver Richtung (d.h. zu höheren Adreßwerten) möglich.

Pseudobefehl GLCH

Syntax: [<Konstante>] GLCH <GLCH-Name>
<GLCH-Name>:= <Name>

Der Pseudobefehl GLCH definiert einen Gleichsetzungsnamen, dem der Wert der Konstanten zugeordnet wird. Fehlt die Konstantenangabe, wird ihm die um 1 erhöhte aktuelle Ablageadresse zugewiesen. Der GLCH-Name kann anschließend als Ersatz für die Ablageadresse benutzt werden (insbesondere für die Pseudobefehle BASIS und SBASIS). Sein Wert wird nicht translatiert.

Pseudobefehl BASIS

Syntax: { <Konstante> / <GLCH-Name> } BASIS

Der Pseudobefehl BASIS gibt dem Assembler die Anweisung, sämtliche Adressen (Ablage- und Folgeadressen) um den Wert der Konstanten oder des GLCH-Namens zu erhöhen. Der Gleichsetzungsname muß vorher definiert worden sein.

Pseudobefehl SBASIS

Syntax: { <Konstante> / <GLCH-Name> } SBASIS

Der Pseudobefehl SBASIS gibt dem Assembler die Anweisung, die Folgeadresse im nächsten Befehl, unabhängig von der eigenen Basis dieses Befehls, mit der durch die Konstante oder durch den GLCH-Namen bestimmten Wert zu translatieren.

Anwendungsbeispiele:

Spalten-Nr.	1 - 6	8 - 13	15 - n	35 - 41
	100	GLCH	MODUL 1	
	200	GLCH	MODUL 2	
	MODUL 1	BASIS		
	0	A	.....	60
	MODUL 2	SBASIS		
	1	B	.....	20
	2	X	.....	64
entspricht	100	A	.....	160
	101	B	.....	220
	102	X	.....	164

Anmerkung: Der Gleichsetzungsname (z.B. MODUL 1) darf max. 6 Zeichen lang sein.

2.4. Kommentare

Kommentare dienen lediglich zur Dokumentation des Programms. Sie werden vom Assembler nicht interpretiert erscheinen aber im Übersetzerprotokoll. Es wird zwischen selbständigen und begleitenden Kommentaren unterschieden.

Selbständige Kommentare werden durch den Buchstaben C in Spalte 1, gefolgt von einem Leerzeichen, eingeleitet. Begleitende Kommentare stehen im Anschluß an eine Informationseinheit (siehe 2.5.).

2.5.
Format-Vorschriften

Für die Assemblersprache WSP 430 ASS wurden folgende Format-Vorschriften festgelegt:

Format einer IE:

Spalten	Inhalt
1 - 6	Ablageadresse der IE in Dezimalziffern (Wertebereich: 0 - 4095) oder in Oktalziffern (Wertebereich: '0' - '7777')
8 - 13	Symb. Name der ALU-Operation
15 - 16	Symb. Name für den A-Operand (AT)
18 - 19	Symb. Name für den B-Operand (BT)
21 - 22	Symb. Name für das Zielregister (DT)
24 - 29	Konstantenteil als Oktalzahl (Wert: '0' - '377') oder als Dezimalzahl (Wert: 0 - 255)
31 - 33	Symb. Name für MSL (Merker setzen/löschen)
35 - 41	NB-Teil, Angabe der 1. Adresse des Viererblocks für den nächsten Befehl durch Oktalzahl (Wert: '0' - '7774') oder Dezimalzahl (0 - 4092) *)
43 - 45	Symb. Name für BSH-Abfrage
47 - 49	Symb. Name für BSN-Abfrage
52 - Ende der Zeile	begleitender Kommentartext (evtl. mit Zeilennumerierung) oder leer

Die übrigen Spalten gelten als Abgrenzung der einzelnen Bausteine der IE und sind leer.

Format der Pseudobefehle:

Spalten	Inhalt
8 - 13	Pseudobefehl
14 - Ende	Kommentar/leer

Format des selbständigen Kommentars

Spalten	Inhalt
1	Buchstabe C (Comment)
2	Blank
Rest	Kommentartext

*) im derzeitigen Ausbauzustand des MIP-Speichers nur bis 2044 bzw. '3774' möglich PE

Pseudobefehle stehen in den Spalten 8-13, der Rest der Zeile wird nicht interpretiert.

Selbständige Kommentare müssen durch die Zeichenfolge C _ in den Spalten 1 und 2 eingeleitet werden, damit sie sich in der 2. Ausbaustufe von einer symbolischen Adresse unterscheiden.

2.6. Optimierung und Speichervergabe

Da die Informationseinheiten eins zu eins abgebildet werden und der Programmierer die Ablageadresse selbst bestimmt, bleiben dem Assembler zur Zeit (in der ersten Ausbaustufe) keine Möglichkeiten zu optimieren oder die Speichervergabe selbst vorzunehmen.

Fehlt die explizite Angabe der Ablageadresse in einer IE, wird die IE unmittelbar hinter die zuletzt bearbeitete abgelegt.

MIBAP-Bit	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Feld-Name	PA	BT						AT					BT			DT												P1	MSL				NB									BSN								F2

Adressbits

Ergänzung auf gerade Bitanzahl

ALU-Operation / Spezial-Operation

Cd	Funktion	Name	UTM
37	A	A	O
32	B	B	O
20	$\overline{A}$	NA	O
25	$\overline{B}$	NB	O
26	A ⊕ B	AUT	O
31	$\overline{A \oplus B}$	NAUT	O
36	A∨B	VEL	O
21	$\overline{A \vee B}$	NVEL	O
33	A∧B	ET	O
27	$\overline{A \wedge B}$	X	O
22	$\overline{A \wedge B}$	Y	O
40	A + MC 1)	AP	Ü
14	A + A	SHL	Ü
11	A + B	ADD	Ü
51	A + B + MC 1)	ADDP	Ü
17	A - 1	AMIN	V
6	A - B - 1	SBMIN	V
46	A - B - 1 + MC 1)	SB	V

Spezial-OT'S DSP-BT = LE *)

	A) KB = DSP-Adresse		
2) 65	Reg → DSP	DBSKB	-
3) 66	DSP → Reg	DBLKB	-
	B) HN = DSP-Adresse		
2) 75	Reg → DSP	DBSHN	-
3) 76	DSP → Reg	DBLHN	-

Spezial-OT'S WSP-ECC

60	Normieren bei ECC-Fehler	NFECC	-
61	Laden d. FE-Bed. nach SC	LFB	-
62	Shiften P0	SHP0	-
63	Shiften P1, P2, P3	SHP123	-

Sonstige ALU-Operationen

34	255	ZP	O
3	-1 (=-255)	ZMIN	O
43	-1 + MC	ZN	V
24	$\overline{A}$	NET	O
41	(AvB) + MC	VELP	Ü
30	$\overline{A \vee B}$	V	O
35	$\overline{A \wedge B}$	W	O
42	A∧B + MC	WP	Ü
13	AB - 1	ETMIN	V
7	$\overline{AB}$ - 1	XMIN	V
54	A + A + MC	SHLP	Ü
10	A + AB	ADET	Ü
50	A + AB + MC	ADETP	Ü
4	A + $\overline{AB}$	ADX	Ü
44	A + $\overline{AB}$ + MC	ADXP	Ü
15	A + (AvB)	ADVL	Ü
55	A + (AvB) + MC	ADVLP	Ü
16	A + ($\overline{AvB}$)	ADW	Ü
56	A + ($\overline{AvB}$) + MC	ADWP	Ü
12	AB + ($\overline{AvB}$)	ETADW	Ü
52	$\overline{AB}$ + ($\overline{AvB}$) + MC	ETADWP	Ü
5	$\overline{AB}$ + (AvB)	VLADX	Ü
45	$\overline{AB}$ + (AvB) + MC	VLADXP	Ü

Erklärungen zu ALU-Operationen

⊕	Exkl. Oder	-	undefiniert
v	Disjunktion	O	gelöscht
·	Konjunktion	Ü	gesetzt bei Übertrag
+	Addition	V	Vorz., gesetzt bei Ergebnis >0
-	Subtraktion		

Cd	Name		BT	Name		DT	Name	Verwendung
0	LE	5)	*) LE	5)	R-WS 22	LA	6)	a) KB = B-Operand bei BT = 2
1	AB		AB			AB		
2	LG		KB	7)		LG		b) KB0 = MIBAR 9 KB1 = MIBAR 10 KB2 = MIBAR 11 bei BSN = 17
3	FR		DR	8)		FR		
4	DR	8)				DR	8)	
5	DW	9)				DW	9)	c) KB = DSP-Adresse bei BT05/06
6	PE					PE		
7	IS					IS		
10	MA				R-WS 23	MA		
11	BP					BP		
12	HC					HC		
13	HD					HD		
14	KA	10)				KR	11)	
15	FC	13)				LE 4) M	12)	
16	FS	14)				FS	14)	
17	MS					MS		
20	HE				R-WS 24	HE		
21	PL					PL		
22	HF					HF		
23	HG					HG		
24	HI					HI		
25	HK					HK		
26	FT	20)				Z1	15)	
27	-					Z2	16)	
30	HM				R-WS 25	HM		
31	HN	17)				HN		
32	HB					HB		
33	HP					KP		
34	MB	18)				FC	13)	
35	ME	19)				FT	20)	
36	SC	21)				ST	22)	
37	SP	23)				-		

Cd	Name		24)	Verwendung
0	0			— 1. Adresse jedes Viererblocks Sie muß durch 4 teilbar sein.
1	LM0			
2	SM0		1	
3	LM1			
4	SM1	25)	2	
5	LM2	26)		
6	DPL	27)		
7	LM3	28)		
10	SM3	29)	10	
11	LM4	30)		
12	LM5			
13	SM5		40	
14	LM6			
15	SM6		100	
16	LM7	31)		
17	SM7	32)	200	
	SMD	37)	20	
	SMU	37)	4	

Cd	Name		Name	
0	SH0		SN0	
1	SH2		SN1	
2	SM0		SM1	
3	ILN	33)	SM3	
4	SM0		SM5	
5	SM6		SM7	
6	BP0		BP1	
7	BP2		BP3	
10	BP4		BP5	
11	BP6		BP7	
12	UTM		SNM	
13	STM		ZM1	
14	FM1		SPM	
15	SNM		EM1	
16	TFK	34)	SMU	
17	KMA	35)	AMA	36)

Bedingter Sprung

Folgeadresse MIBAR, Bit

Merker Setzen/Löschen

Parity MIBAP 22-43

Parity MIBAP 2-23

*) bei allen DSP-Operationen

1) Für alle Funktionen + MC gilt

MC vorher	Übertrag	Ergebnis
gelöscht	nein	+0
gelöscht	ja	+1
gesetzt	nein	+1
gesetzt	ja	+1

2) Quellenregister = AT-Feld

3) Zielregister = DT-Feld

4) Manuelles Laden über SDEFR, wobei Bitmuster in SSTA. Selektierung durch SRWO

5) Leer = Pseudoregister mit Dauerinhalt "0"

6) Lampenanzeigeregister

7) Konstante (MIBAP 22-29)

8) WSP-Daten bei
A) Lesen: DR-DW
B) Schreiben: KA-DR(DW).
Wird bei automatischer Datenphase überschrieben. Speicherung des zuletzt aufgeschriebenen Bytes im DR, des vorletzten im DW

9) Datensender zum Rechner. (Lesen/Schreiben siehe 8)

10) Datenempfang vom Rechner. Durchschaltbed.: STP

11) Quittierung

STP	AF	AMP	GA	FE	IE	AN2	AN1

12) Stellung des SRWO bei manuellem Merker-Setzen.

13) WSP-BUSOUT-Register

14) ORIIS-: WSP-BUSIN-, Geratefehler- u. Modul-Adreßregister. Selektierung über FT6,7

15) Datenlängenzähler 2^0-2^7

16) Datenlängenzähler 2^8-2^{15}

17) DSP-Adr.-Register bei BT75/76

18) WSP-ECC Fehlererkennung 1 bei Lesen

19) WSP-ECC Fehlermuster

20) FT0-4: WSP-TAGBUS-Register FT6,7: Selekt. d. FS-Eing.

FT:	7	6
BUSIN → FS	0	0
Geratefehler → FS: FS0: Modul-Select-FE 1: Parity-FE BUSIN 2: Parity-FE BUSOUT 3: TAG Invalid 4: -- 5: Device Check	0	L
FS0-5: Mod.-Adr. 3/6 Cd. FS6,7: AW-Adresse	L	0
DBUS-Zielregister-Freigabe	L	L

21) WSP-ECC Fehlererkennung 2 bei Lesen und Schreiben

22) Spez. Steuermeldungen
ST0,1,4 für WSP-Lese- und Schreibbetrieb
ST5,6,7 für WSP-ECC-Logik

23) Programm-Stop (kein Register)

24) Einst. am SSTA für manuelles Setzen über SDEFR, wobei SRWO in Stellung "M".

25) Setzbed.: MSL4-Index

26) Löschen Uhrmerker
Setzen erfolgt nach 25ms oder bei Übertragungsfehler
Abfragbar durch BSN16.

27) = Übernahmetakt für Laden des Diagnostik-Programms in den MIP-Speicher.
Taktfolge: 2,2µs-130µs
Nach dem 10. DPL 1 Leerbefehl notwendig.

28) Löschbed.: MSL7 oder bei DPL durch Hardwarefehler

29) Setzbed.: MSL10 oder MIBAP47: UTM

30) Merker 0 wird bei Zeichenübertragung von und zum WSP automatisch gesetzt und muß durch MSL11 gelöscht werden.

31) Löschbed.: MSL16 oder bei Ende der Datenübertragung

32) Definiert den Beginn der automatischen Datenübertragung. Hierbei wird DR-, DW und KA-Register überschrieben

33) ILN, wenn (SB auf INUNE) [(STIMA auf BUNDEL) oder (STIMA auf FEHLER LA = Null)]

34) Rechner-Doppelzugriff
TFK: Rechner 1,
$\overline{TFK}$: Rechner 2

35) KB0 = MIBAR 9
KB1 = MIBAR 10
KB2 = MIBAR 11

36) ABUS = MIBAR 0-7

37) Nur manuell- und durch spez. Hardware-Bedingungen (z. B. Byte-Merker oder Hardware-Fehler) setzbar

38) Die Steckenhertenzuordnung der Register bezieht sich auf den A-Operanden.

Mikrobefehlsformat WSP 430

3. FUNKTIONSBESCHREIBUNG

3.1. Einleitung

Die Funktionen des Assemblers PS& WSP 430 ASS sind im wesentlichen die Abbildung der Eingabequelle in den Ausgabecode, die Erstellung eines Übersetzerprotokolls, die Ausgabe einer Crossreferenzliste und die Ausgabe einer Speicherbelegungsliste.

3.2. Eingabequelle

Die WSP 430 ASS-Quelle kann im Batchbereich über Lochkarten als Fremdstring, von Magnetband oder aus der LFD als Texthaltungsdatei eingegeben werden. Im Gespräch ist die Eingabe als Fremdstring von Konsole oder Eingabe als Datei zugelassen. In jedem Fall darf eine Eingabeeinheit (Lochkarte, Satz, Zeile) 160 Zeichen nicht überschreiten. Die minimale Länge einer codeerzeugenden IE ist durch die Formatvorschriften für die Assemblersprache WSP 430 ASS auf 52 Zeichen festgelegt (siehe 4.2.5.). Die Mindestlänge für selbständige Kommentare beträgt 2, für Pseudobefehle 13 Zeichen.

3.3. Ausgabecode

Der Assembler PS& WSP 430 ASS erzeugt auf Wunsch des Benutzers Ausgabecode nach vorgeschriebenem Format. Jedes CODE-Element besteht aus der absoluten Ablageadresse und der Interndarstellung eines Mikrobefehls. Der Aufbau der Mikrobefehle ist in 2.2.1.1. beschrieben.

Der erzeugte Code wird in eine Datei namens WSP 430 CODE der Standarddatenbasis &STDDB des TR 440-Systems TNS 440 ausgegeben. WSP 430 CODE ist eine sequentielle Datei, ein Satz ist genau 20 Ganzworte (TR 440-GW) lang. Das entspricht dem Speicherbedarf für 12 CODE-Elemente. Diese Datei kann mit den gebräuchlichen Kommandos des TNS 440 weiterverarbeitet werden. Üblicherweise wird die Datei mit Hilfe des Programms MIKROCAS (s. Kap. 7) auf Magnetband-Kassette ausgegeben, da diese als Eingabemedium für das Anpaßwerk WAW 431 bzw. PAW 405 verwendet wird.

3.4. Übersetzerprotokoll

Das Übersetzerprotokoll wird je nach den Angaben des Benutzers mit oder ohne Objektcode, ein- oder zweizeilig, auf Konsole oder Drucker oder garnicht ausgegeben.

Erkennt der Übersetzer schwere syntaktische Fehler, wird kein Ausgabecode erstellt. Die Fehlermeldungen und Warnungen werden deutlich sichtbar in Klartext unmittelbar unter der Zeile mit der falschen IE protokolliert. Sie werden mit fünf Pluszeichen eingeleitet.

Fehlerhafte IE'n werden grundsätzlich (auch bei abgeschaltetem Protokoll) protokolliert.

Format einer Protokollzeile mit Objektcode:

Spalten 1- 6: sechsstellige laufende Zeilennummerierung (dezimal, ohne führende Nullen)

Spalten 9-14: Ablageadresse (rechtsbündig als Oktalziffern mit führende Nullen)

Spalten 16-53: Objektcode (in Oktalziffern, siehe 3.4.1.)

Spalten 55-n: Protokoll der Quellzeile

Format einer Protokollzeile ohne Objektcode:

Spalten 1-6: laufende Nummer

Spalten 9-n: Protokoll der Quellzeile

Über jede neue Seite wird eine Kopfzeile ausgegeben.

Kopfzeile für Protokollierung mit Objektcode:

"NUMER. ␣␣ AB-ADR ␣ P ␣ OP ␣ AT ␣ B ␣ DT ␣ KON ␣ P ␣
SL ␣␣ NB ␣ NBA ␣ SH ␣ SN ␣ P ␣␣
AB-ADR ␣ ALU-OP ␣ AT ␣ BT ␣ DT ␣ KONST. ␣ MSL ␣ NB-TEIL ␣
BSH ␣ BSN ␣␣ KOMMENTAR"

Kopfzeile für Protokollierung ohne Objektcode:

"NUMER. ␣ AB-ADR ␣ ALU-OP ␣ AT ␣ BT ␣ DT ␣ KONST. ␣
MSL ␣ NB-TEIL ␣ BSH ␣ BSN ␣␣ KOMMENTAR"

Die Kopfzeilen entsprechen dem Aufbau der Protokollzeilen auf dem Drucker. Sie erscheinen nicht im Konsolprotokoll.

3.4.1.
Protokollierung des Objektcodes

Der Objektcode wird durch 26 Oktalziffern dargestellt. Die Bedeutung der Oktalziffern wird durch die folgende Tabelle verdeutlicht:

Oktal-Ziffer *)	Bereich	Bedeutung
1	0/1	Paritybit PA für die Ablageadresse
2 3	0-7 0-7	ALU-Operation
4 5	0-3 0-7	A-Operand (AT)
6	0-3	B-Operand (BT)
7 8	0-3 0-7	Zielregister (DT)
9 10 11	0-3 0-7 0-7	Konstantenteil
12	0/1	Paritybit P1 für Bits 22-48
13 14	0-1 0-7	MSL-Teil
15 16 17	0-1 0-7 0-7	NB-Teil (seitenrelative Viererblocknummer)
18 19 20 21	0 0-7 0-7 0-7	NBA (seitenrelative Absoluteadresse des nächsten Viererblocks)
22 23	0-1 0-7	BSH-Teil (bedingter/unbedingter Sprung)
24 25	0-1 0-7	BSN-Teil (bedingter/unbedingter Sprung)
26	0/1	Paritybit P2 für die Bits 2-20 des Mikrobefehls

*) Oktal-Ziffer von links nach rechts durchnumeriert

3.4.2.
Ausgabe der Fehlertexte

Fehlerhafte Informationseinheiten werden gründsätzlich protokolliert. Die folgende Tabelle enthält die möglichen Fehlertexte.

Falscher Eingabecode*	ALU-Operation fehlt
Falsche Zeichenfolge*	AT-Teil fehlt
Name hier verboten*	BT-Teil fehlt
Keine ALU-Operation	DT-Teil fehlt
Keine MSL-Anweisung	Konstante fehlt
Kein BSH-Teil	MSL-Teil fehlt
Kein BSN-Teil	NB-Teil fehlt
Dezimalzahl hier verboten*	BSH-Teil fehlt
Ablageadresse zu groß	BSN-Teil fehlt
BT-Teil falsch	In Ablageadresse
Konstante zu groß	In ALU-Operation
MSL-Teil falsch	In AT-Teil
NB-Teil zu groß	In BT-Teil
Warnung: keine VB-Adresse	In DT-Teil
Oktalkonstante hier verboten*	In Konstantenteil
Kein A-Operand	In MSL-Teil
Kein B-Operand	In NB-Teil
Kein D-Operand	In BSH-Teil
Adresse doppelt belegt	In BSN-Teil
Blank fehlt*	IE UNVØLLSTAENDIG

Anmerkung: Die mit * versehenen Fehler werden nur in Zusammenhang mit einem der Fehler 33-45 ausgegeben.

Beispiel: 490 '118' DBSKO .....etc.

```
+++++ FALSCHE ZEICHENFOLGE
+++++ IN ABLAGE-ADR
```

3.5. Übersetzung des Quellprogramms

Die Mikrobefehle werden an explizit angegebener Adresse in einem Bereich abgelegt. Im Grunde würde deshalb ein Assemblerlauf für die Syntaxanalyse und gleichzeitige Codeerzeugung genügen. Da aber einerseits die symbolische Adressierung in Aussicht steht, andererseits bestimmte Fehlerfälle nicht erkannt würden (z. B. Folgebefehl nicht definiert), sollen die einzelnen Funktionen von vornherein getrennt bearbeitet werden.

Zu diesen Funktionen zählen:

- die syntaktische Analyse,
- die Objektcode-Erzeugung,
- die Erstellung des Übersetzungsprotokolls,
- die Erstellung der Crossreferenzliste und
- die Ausgabe der Speicherbelegungsliste.

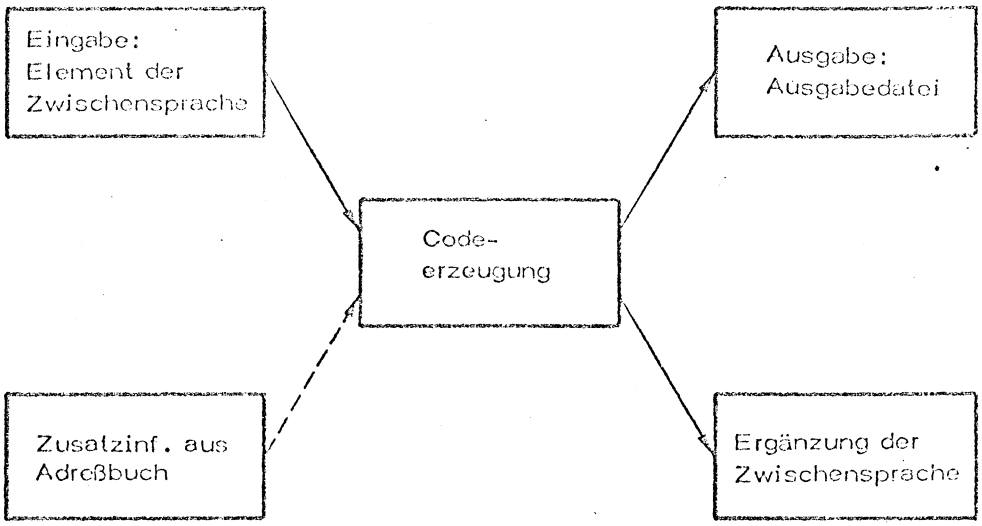
3.5.1. Syntaktische Analyse

Bei der syntaktischen Analyse wird eine Quellzeile nach der anderen eingelesen und auf Syntax-Fehler untersucht. Dabei wird die Einheit gleichzeitig soweit wie möglich auf ihre interne Darstellung abgebildet.

Der vorläufige Ausgabecode, die Fehlerkennzeichen für etwa vorhandene Fehler und die Quellzeile werden in einer Zwischensprache deponiert.

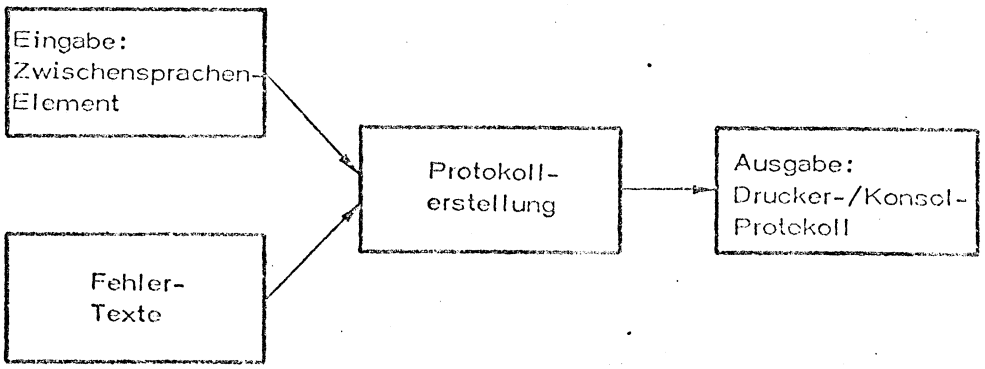
3.5.2.
Objektcode-Erstellung

Bei der Objektcode-Erzeugung wird ein Element der Zwischensprache nach dem anderen verarbeitet. Der vorläufige Ausgabecode wird ergänzt (mindestens um die Paritybits) und in die Ausgabedatei WSP 430 CODE geschrieben. Die Kennzeichen der evtl. auftretenden Fehler werden in die Zwischensprache übernommen.



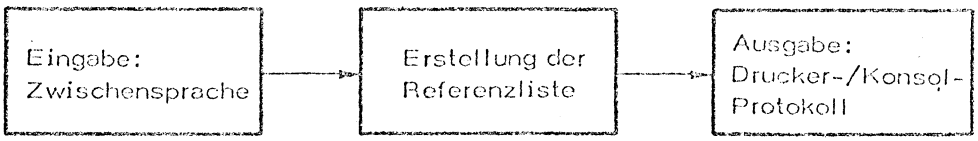
3.5.3.
Erstellung des
Übersetzungsprotokolls

Die Protokollfunktion wertet alle Angaben des Benutzers zur Protokollgestaltung und die protokollbetreffenden Pseudobefehle aus. Fehlerhafte Informationseinheiten werden auf jeden Fall ausgegeben. Als Eingabeeinheiten dienen die Elemente der Zwischensprache. Sie werden sequentiell abgearbeitet, so daß die Quellzeilen in unveränderter Reihenfolge im Assemblat erscheinen.



3.5.4.
Erstellung der Cross-referenzliste

Bezugspunkte der Crossreferenzliste sind in der ersten Ausbaustufe die Anfangsadressen der durch das Programm belegten (oder teilweise belegten) Viererblöcke. Die Adressen werden durch Oktalzahlen dargestellt (Bereich: '0' - '7777'). Die Referenzen werden durch die laufende Quellzeilennumerierung angegeben. Es werden maximal 10 Referenzen pro Zeile gedruckt. Für die Ausgabe der Referenzen wird ein Teil der Zwischensprache, eine Liste mit den Code-Elementen, ausgewertet.



Bei bedingten Sprüngen wird die Referenz nur bei der Adresse angegeben, die bei nicht erfüllter Bedingung angesprungen wird. Bei AMA-Sprüngen (siehe 2.2.1.7.2. und 2.2.1.7.4.) wird die Referenz dort aufgeführt, wohin mit <AT> = 0 gesprungen würde.

Das Protokoll der Referenzliste wird nach folgendem Format ausgegeben:

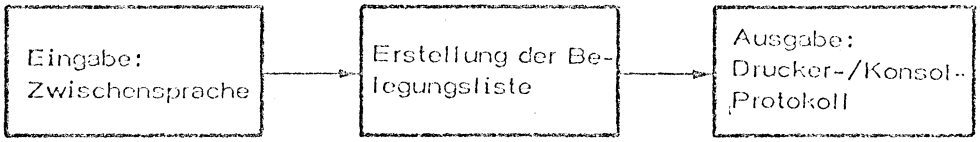
Spalten-Nr.	Inhalt
2 - 5	Ablageadresse des Mikrobefehls (Bezugspunkt) in Oktalziffern mit führenden Nullen
9	Buchstabe "D", für Definitionsort
12 - 17	Zeilennummer des Definitionsortes, maximal 6 Dezimalziffern ohne führende Nullen
21	Buchstabe "R" für Referenzorte
24 - n	Angabe der Referenzorte durch 6-stellige Zeilennummern (maximal 10 Angaben pro Zeile). Benachbarte Zeilennummern werden durch je zwei Blanks von einander getrennt.

Sollten mehr als 10 Referenzen zu einem Bezugspunkt gehören, werden sie in der nächsten Zeile ab Spalte 24 ausgegeben.

3.5.5.
Erstellung der Speicherbelegungsliste

Für den Programmierer des Anpaßwerkes ist eine Speicherbelegungsliste von besonderer Bedeutung. Da die Befehle nicht sequentiell abgelegt werden können, entstehen oftmals Lücken in der Speicherbelegung. Damit der Programmierer den Verschnitt möglichst klein halten kann (notfalls können die Lücken durch ein anderes Programm belegt werden), gibt der Assembler folgende Daten aus:

Anzahl der belegten Speicherzellen:	1 (Dezimalzahl)
erste belegte Adresse:	'm' (Oktalzahl)
letzte belegte Adresse:	'n' (Oktalzahl)
unbelegte Adressen:	'n1' - 'n2'
	'n3' - 'n4' usw.



In der Belegliste wird folgende Information ausgegeben:

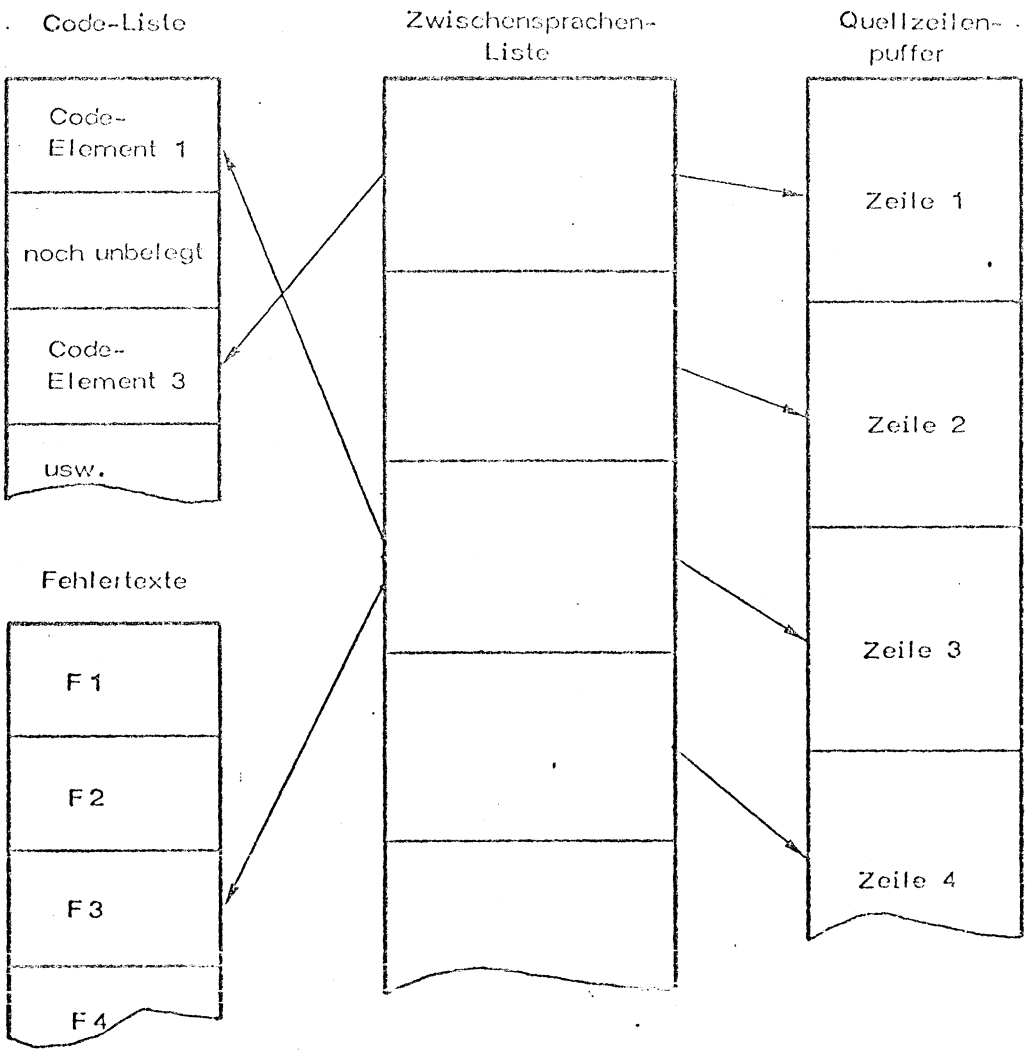
Anzahl der belegten Speicherzellen:	1 (Dezimalzahl)
erste belegte Adresse:	'm' (Oktalzahl)
letzte belegte Adresse:	'n' (Oktalzahl)
nicht belegte Adressen	'n1' - 'n2'
	'n3' - 'n4'
	usw.

3.6. Zwischensprache

Die Zwischensprache wird vom Analyselauf aufgebaut. Dabei kann der größte Teil des endgültigen Ausgabecodes erstellt werden. Um dieses Zwischenergebnis nicht ein zweites Mal berechnen zu müssen, sollte es in der Zwischensprache gesichert werden. Für die fehlenden Werte werden Verweise in die Zwischensprache übernommen. Auf diese Weise wird bei der Codegenerierung eine zweite Analyse der Quelle vermieden. Da eine Quellzeile und der zugehörige Interncode in einer Protokollzeile ausgedruckt werden sollen, müssen die Quellzeilen notgedrungen zwischengespeichert werden.

Fehler sollen unmittelbar hinter den falschen Einheiten gemeldet werden. Aus diesen Gründen sollte die Zwischensprache ebenfalls Verweise auf den Anfang der Quellzeilen, Verweise auf den zugehörigen Interncode und Verweise (Fehlerschlüssel) auf die Texte der zu meldenden Fehler enthalten. Durch die Aufnahme von Verweisen in die Zwischensprache ist gewährleistet, daß die Elemente der Zwischensprache eine einheitliche Länge haben. Die Zwischensprache selbst zerfällt aber in mehrere Teile: in eine zentrale Liste, in eine Liste mit den Code-Elementen, in einen Puffer für die Quellzeilen. Die Liste der Code-Elemente wird so aufgebaut, daß die Reihenfolge der Elemente in etwa die Speicherbelegung abbildet. Auf diese Weise macht die Code-Liste erstens eine Aussage über die Speicherbelegung, und zweitens kann sie zur Erstellung der Crossreferenzliste herangezogen werden.

Der Zusammenhang zwischen den einzelnen Listen soll durch folgende Zeichnung veranschaulicht werden:



Der Aufbau der einzelnen Listen-Elemente bleibt der Programmkonstruktion überlassen.

4. DATEISPEZIFIKATION

Für den Ausgabecode kreiert der Assembler in der Standarddatenbasis &STDDB des TNS 440 eine sequentielle Datei "WSP 430 CODE". Die Satzzahl dieser Datei wird mit ungefähr 350 Sätzen angegeben, der Satzbau mit genau 20 Worten. Die Datei ist so bemessen, daß genügend Platz für die maximale Anzahl von Informationseinheiten (4096 IE'en) vorhanden ist.

Für die Listen der Zwischensprache werden vom Assembler Kernspeicher und Hintergrund-Gebiete erstellt.

5. KONSTRUKTIONS-RANDBEDINGUNGEN

5.1. Anforderungen an die Gerätekonfiguration

Der Assembler PS&WSP 430 ASS ist nur auf TR 440-Anlagen lauffähig, die mindestens folgende Peripheriegeräte haben:

Hintergrundspeicher (für Ausgabedatei und Zwischensprachgebiete)
Lochkartenleser (für die Eingabe des Abschnittes)
Schnelldrucker (für Protokollausgabe)
Bandkassetten-Beschriftungsgerät (für die Ausgabe des erzeugten Codes mit Hilfe des Programms MIKROCAS, s. Kap. 7)

Die Größe des Assemblers wird so bemessen sein, daß er auch auf TR 440-Anlagen mit kleiner Kernspeicherkapazität ohne Laufzeitverlängerung arbeiten kann.

5.2. Anforderungen an die Software-Schnittstelle

Der Assembler PS&WSP 430 ASS soll unter dem TNS 440 laufen. Aus der Datenbasis &QEFDB benötigt er folgende Dienstunterprogramme:

S&QPZEIT	(für Start- und Endemeldung des Assemblers)
S&ALARM	(für Alarmbehandlung)
S&SRF	(für Behandlung der SSR-Fehler)
S&GZF	(für das Einlesen der Quelle)
S&SMFEHL	(für S&GZF-FEHLER-Behandlung)

Vom Abwickler und von der Datenbasis werden folgende Leistungen verlangt:

SSR 1 4	(Lesen der Steuerinformation)
SSR 3 0 SSR 3 4 SSR 3 36 SSR 3 68	(Gebietsdienste für die Zwischensprache-Gebiete)
SSR 253 3 SSR 253 9 SSR 253 10 SSR 253 11	(Datenbasisdienste für die Bearbeitung der Ausgabedatei)
SSR 0 12 SSR 0 16	(zum Beenden des Operatorlaufs)
SSR 6 0 SSR 6 8 SSR 6 12 SSR 6 16	(Protokolldienste)

Vom Entschlüssler werden die Leistungen des Kommandos UEBERSETZE verlangt.

PE

6. ZUSATZPROGRAMM WSP&SCHIEBER

Das Programm WSP&SCHIEBER ermöglicht eine Verschiebung der Mikrobefehls-Ablageadressen sowohl in positiver Richtung (= höhere Adressen) als auch in negativer Richtung (= niedrigere Adressen). Somit besteht die Möglichkeit, mehrere Programme (z.B. Testprogramme) zu ketten, wobei auch Programme mit ursprünglich hohen Adressen (z.B. Diagnostik-Programme) gekettet werden können.

6.1. Benutzungsanleitung von WSP&SCHIEBER

Der Operator WSP&SCHIEBER liegt in einer Datei WSP&SCHIEBER der UFD mit dem Benutzerkennzeichen EP65A.
Er wird durch die Kommandofolge

```
□ LFANMELDE,LESEN=EP65A.WSP&SCHIEBER
□ BINAEREIN,INFORMATION=WSP&SCHIEBER
□ LFABMELDE,DATEI=WSP&SCHIEBER
```

in die Standard-Datenbasis &STDDB geladen und mit

```
*<VZ> ZZZZLL Kommentartext
```

```
□ STARTE,PROGRAMM=WSP&SCHIEBER,
  DATEN=/<WSP430-QUELL-PROGRAMM>
```

gestartet.

Liegt das Quellprogramm in einer Datei, muß das STARTE-Kommando heißen:

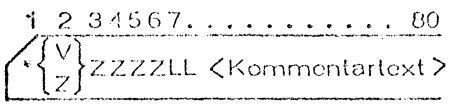
```
□ STARTE,PROGRAMM=WSP&SCHIEBER,
  DATEI=98-<Dateiname der WSP430-Quelle>
```

Das Programm WSP&SCHIEBER erbringt folgende Leistung:

Jede Quellzeile wird in den Positionen "Ablageadresse" und "NB-Teil" um den aktuellen Verschiebungsfaktor erhöht. (Eine Prüfung, ob in gewissen Fällen die Position "KONSTANTE" verändert werden muß, findet nicht statt.)

Der aktuelle Verschiebungsfaktor wird durch eine Anweisung vor dem betreffenden Quellbereich angegeben.

Aufbau der Anweisung:



Spalte 1 der Anweisung enthält ein Sternchen (*).
Spalten 2-6 enthalten eine Dezimalzahl und zwar rechtsbündig (ggf. mit einem Vorzeichen (+/-)).
Führende Nullen dürfen durch Blanks ersetzt werden.

Das WSP-Quell-Programm wird in veränderter Form in eine Datei mit dem Namen DATEI eingetragen. Dieses "verschobene" Quellprogramm kann durch das Kommando:

```
STANZE,GERAET=KS-KC1,
  INFORMATION=DATEI
```

auf Lochkarten ausgestanzt werden oder durch die Kommandofolge:

```
LFAN.,EP65A.PS&WSP430ASS
BINAEREIN,PS&WSP430ASS
LFAB.,PS&WSP430ASS
UEBERSETZE,QUELLE=DATEI,SPRACHE=TAS, etc.
```

direkt aus der Datei DATEI vom WSP430-Assembler übersetzt werden. Dabei ist zu beachten, daß die Datei im 2. Fall nicht automatisch gelöscht wird. Es empfiehlt sich, nach jeder Übersetzung die Datei mit dem LOESCHE-Kommando zu löschen:

```
LOESCHE,DATEI=DATEI
```

7.4.
Definition des Kommandos CASAUS

Das Tätigkeitskommando CASAUS wird durch das folgende DEFINIERE-Kommando in das Entschlüsselgedächtnis eingetragen.

```
Δ DEFINIERE
      ,TAETIGKEIT = CASAUS
      ,OPERATOR   = MIKROCAS
      ,SPEZIE     = DATEI (NL, SN)
                    ANZAHL (NL, NZ4)
                    ZUSATZ (NL, N)
      ,OBLIGAT    = 1
```

7.5.
Kommandobeschreibung

CASAUS	Beschriften von Magnetbandkassetten

Spezifikation

(1)	DATEI	Bezeichnung der Eingabedatei
(2)	ANZAHL	Anzahl der Ausgabevorgänge
(3)	ZUSATZ	Zusätzliche Angaben zur Dateibearbeitung

Kommando für Programmiersystem

Wirkung:

Werden nur Angaben unter DATEI gemacht, wird der Inhalt der Datei in einem Satz abgelegt und 2x auf Kassette ausgegeben. Durch die Spez. ANZAHL kann die Zahl der Ausgabevorgänge gesteuert werden. Werden Angaben unter ZUSATZ gemacht, werden die ersten beiden Sätze der Eingabedatei überlesen.

DATEI	Bezeichnung der Eingabedatei

spez. Wert

datei	:	Name der Datei (Standardname)

obligate Spezifikation zum Kommando CASAUS

Die Datei, die in der Standarddateibasis liegen muß, enthält die auszugebende Information. Mit Hilfe des Operators MIKROCAS werden die Sätze der Eingabedatei zu einem Satz zusammengefaßt. Die Eingabedatei darf max. 8K TR440-Ganzworte lange sein. Die Datei muß sequentiell aufgebaut sein und einen Satzbau von G20W haben.

ANZAHL	Anzahl der Ausgabevorgänge
n	Die Information soll n-mal ausgegeben werden
2	Voreinstellung

optionale Spezifikation zum Kommando CASAUS

Die Information wird n-mal ausgegeben, maximal jedoch 16 mal. Wird eine Zahl größer 16 angegeben, wird 16 eingesetzt. Nach jeder Ausgabe erfolgt ein Kassettenauswurf.

ZUSATZ	Zusätzliche Angabe zur Dateibearbeitung
-	keine Zusatzangabe (alle Sätze der Eingabedatei werden verarbeitet)
UEB	Die ersten 2 Sätze der Eingabedatei werden überlesen

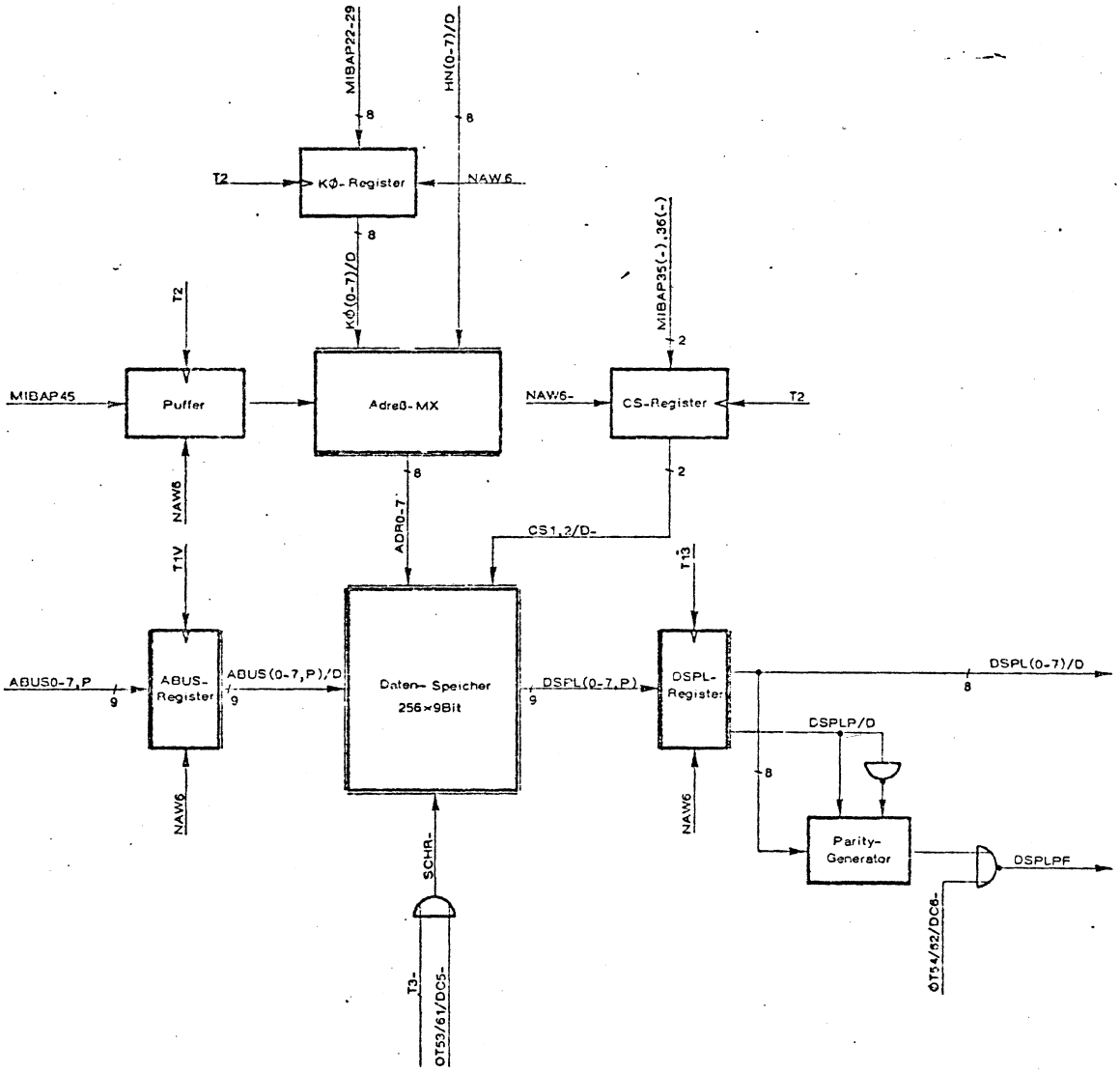
optionale Spezifikation zum Kommando CASAUS

Werden keine zusätzlichen Angaben zur Dateibearbeitung gemacht, werden alle Sätze der Eingabedatei bearbeitet. Bei der Angabe UEB werden die ersten zwei Sätze der Eingabedatei überlesen.

7.6. Anwendungsbeispiel

```
Δ CASAUS
  ,DATEI   = WSP 430 CODE
  ,ANZAHL  = 8
  ,ZUSATZ  = UEB
```

Der Inhalt der Datei WSP 430 CODE wird in einem Block auf 8 Kassettenseiten ausgegeben, wobei die ersten beiden Sätze der WSP 430 CODE überlesen werden.



Blockschaltbild R-WS36

PE

4.5.1-2920

11.4.
Zusatzoperationen
(siehe Übersichtstabelle 5)

Folgende Operationen werden vom AW nur in Verbindung mit Statusabfragen, Lese-Schreib- oder Steuerbefehlen ausgewertet.

11.4.1.
Wartungsmodus:
3. SIZ: Bit $2^2 = 1$

Diese Zusatzoperation ist nur sinnvoll bei

- Lesen HA und R0
- Lesen Ct
- Prüflesen

Tritt bei Lesen HA bzw. Ct korrigierbarer Prüfzeichenfehler auf, so wird der Fehler nicht dem Rechner gemeldet (Normalfall). Die Datenbytes von HA bzw. Ct werden nur korrigiert zum Rechner übertragen.

Ist das Wartungsmodusbit gesetzt, so meldet das AW auch korrigierbare Prüfzeichenfehler zum Rechner. Die Daten werden dann nicht zum Rechner übertragen.

11.4.2.
Offset

In jedem Laufwerk besteht die Möglichkeit mit einer Offset-Einrichtung die Lese-Schreib-Köpfe im Bereich 25 µZoll - 1575 µZoll bei 100 MByte- und 25 µZoll - 775 µZoll bei 200 MByte-Laufwerken aus der Spurmitte in beide Richtungen zu bewegen.

Richtung Plattenmitte	"Offset Vorwärts"
Gegenrichtung	"Offset Rückwärts"

Diese Offset-Operation ist nur durchführbar als separater Positionierbefehl.

Im Normalbetrieb sind folgende Offset-Schritte zu empfehlen: 200 µ", 400 µ", 800 µ"* vorwärts und rückwärts.

Folgende Betriebszustände führen zu Gerätefehler:

- Lese/Schreibköpfe sind in Offset-Position und ein Positionierbefehl (kein Offset-Positionierbefehl) soll gestartet werden.

- Lese/Schreibköpfe in Offset-Position und ein Schreibbefehl soll ausgeführt werden.

Das Offset Bit 4. SIZ: Bit 2^5 wird nur bei Positionieren (siehe 11.2.2) ausgewertet. Bei allen anderen Befehlen wird dieses Bit ignoriert.

Die Offset Schrittweite entnimmt das AW dem 7. SIZ der Startinformation. Ist dieses 7. SIZ gleich Null und das Offset Bit gesetzt, so führt das Laufwerk ein Offset-Normieren durch. Die Lese/Schreibköpfe werden wieder in Spurmitte positioniert. Jedes Offset Positionieren dauert bis zu 5 msec. Ist das Offset-Bit in der Startinformation für Positionieren gesetzt, so wird nur eine Offsetbewegung durchgeführt. Die in der Startinformation angegebene Zylinderadresse wird vom AW nicht ausgewertet.

PE

*) bei WSP 430

Sinnvoller Einsatz der Offset-Einrichtung:

Erkennt das AW einen unkorrigierbaren Prüfzeichenfehler bzw. ist eine Sektoradresse nicht auffindbar, so kann der Lesebefehl in Offset Position wiederholt werden und zwar in 64 verschiedenen Offset-Kombinationen in beiden Richtungen aus der Spurmitte. Nur die großen Offset-Schritte 200- bis 800µ Zoll sind sinnvoll.

Folgende Startinformationsfolge wird für diesen Fehlerfall empfohlen:

- Positionieren nach vorgegebener Zylinderadresse
- Anruf → Statusabfrage: Positionieren Ende (5-50 msec nach Positionieren-Start)
- Lesen, Abbruch Lesen mit Fehlereingriff "Prüfzeichenfehler"
- Statusabfrage bei Fehler
- Offset Positionieren
- Anruf → Statusabfrage: Offset Ende (nach 5 msec)
- Lesen
- Offset Normieren
- Anruf → Statusabfrage: Positionieren Ende, Offset Normiert
- Positionieren nach anderer Zylinderadresse

11.4.3.
Abbruch durch Rechner:
4. SIZ: Bit 2^6 = 1

Diese Zusatzoperation ermöglicht die vollständige Ausschaltung der Blocklängenprüfung durch das AW (Unstimmigkeit bei vorformatierten Sektoren mit Datenblocklängen die kein ganzzahliges Vielfaches von 6 bzw. 7 Bytes sind). Diese Zusatzoperation wird vom AW nur für Lese- und Schreibbefehle ausgewertet (genaueres siehe Pkt. 11.3).

11.4.4.
12 Sektoren/Spur-Modus:
4. SIZ: Bit 2^7 = 0

Im 12 Sektoren/Spur-Modus (definiert durch 4. SIZ, Bit 2^7 = 0) sind nur genau 12 Sektoren pro Spur zulässig. Soll mit anderer Sektoranzahl (bzw. Datenlänge) gearbeitet werden, so ist Bit 2^7 auf L zu setzen. Dann sollte allerdings nur ein Sektor pro Auftrag gelesen/geschrieben werden, da sonst in einem seltenen Fall ein nicht erkennbarer Fehler auftreten kann (dies gilt nicht für W-Testprogramme, da dort über den Informationsvergleich im Rechner dieser Fehler erkannt wird).

Name Zusatzoperation	3. SIZ	4. SIZ	5. - 12. SIZ	Bemerkungen
		Operations-code	5. SIZ = Leerzeichen. 6. SIZ Sektoranzahl 7. SIZ = Offset Operation 8. SIZ = Laufwerksadresse 9.-12. SIZ=Zylinder-Kopf- u. Sektoradresse	
- Wartungsmodus (nur bei Lesebefehlen)	001011XX	Lesen 0X010XXX	5. - 12. SIZ entsprechend obigen Lese-befehlen	Meldung auch von korrigier-baren PfZ-Fehlern bei "Prüf-lesen", Lesen HA und R0 und Lesen Ct.
- Offset	001010XX	00101001 Pos. Offset	5.,6.,8.-12. SIZ entsprechend obigen Lesebefehlen 7. SIZ = X0XXXXXX Offset-Operation Vorwärts/Rückwärts	Nur in Verbindung mit Positionieren vom AW ausgewertet.
- Abbruch durch Rechner	00101XXX 00101XXX	Lesen 01010XXX Schreiben 01011XXX Abbr. durch Rechner	5. - 12. SIZ entsprechend obigen Lese- und Schreibbefehlen	

Es bedeuten X = 0 oder 1, 5. SIZ Leerzeichen immer = 00000000

Befehlstabelle 5: Zusatzoperationen
(können Bestandteil jedes Befehls sein)

12. TESTBETRIEB

(Schnittstelle zur Wartung)

Als Wartungshilfen werden Diagnostik-Routinen bereitgestellt, die die Fehlersuche im AW und in den Laufwerken erleichtern soll. Die Diagnostik-Routinen sind Mikroprogramme, die möglichst alle Funktionen der AW- und Laufwerkselektronik testen. Die einzelnen Diagnostik-Routinen können wahlweise in den MIP-Speicher des AW geladen werden:

- Off-Line Diagnostik-Routinen (siehe 12.1) über die Magnetbandkassette;
- On-Line und In-Line Diagnostik-Routinen (siehe 12.2 und 12.3) über den Rechner über den Befehl "Diagnostik-Programm Laden" (siehe 11.1.4).

12.1. Off-Line Testbetrieb

Im Off-Line Testbetrieb ist die Verbindung Rechnerkanal - AW verriegelt (Betriebsartenschalter des inneren Bedienfeldes SB auf Stellung "Test").

Off-Line Diagnostik-Routinen können nur im Off-Line Testbetrieb laufen, dabei handelt es sich um

- AW-interne Tests wie
  Registertests, Datenspeichertests, Rechenwerkstest, Merker-Test, Test der ECC-Elektronik und der Schnittstelle zu den Laufwerken und
- Laufwerkstests wie
  Positionieren zählend und springend, Offset-Positionieren, Programme zur Justage bzw. Überprüfung der Lese-Schreib-Köpfe und zur Einstellung der Geschwindigkeit der Voice-Coil.

Alle diese Tests würden den Normalbetrieb Rechner - Laufwerk stören und können daher nur im Off-Line Testbetrieb gefahren werden.

Die einzelnen Tests werden über die Startadresse der Mikro-Routinen einstellbar am inneren Bedienfeld gestartet.

Die Off-Line Diagnostik-Routinen belegen den gesamten Mikroprogrammspeicher des AW's, das Betriebs-Mikroprogramm (BMP) wird also überschrieben.

Nach Abschluß dieses Wartungsbetriebs muß daher das BMP über eine Magnetbandkassette neu geladen werden.

12.2. On-Line Testbetrieb

On-Line Diagnostik-Routinen sind Mikroprogramme zum Test der Laufwerkselektronik (Laufwerk mit "Service" Adreßstecker S, siehe Abschn. 2.1).

Im On-Line Testbetrieb können Diagnostik-Routinen mit einem Laufwerk neben dem normalen Plattenspeicherverkehr Rechner - Laufwerk ablaufen. Der Normalbetrieb wird hierdurch zeitlich verzögert.

Die Ablaufdauer einer On-Line Diagnostik-Routine - ein Durchlauf ≥ 100 ms - erfordert eine Rechnerkoordination des Normalbetriebs mit dem Ablauf der Diagnostik-Routine im AW, da sonst der Normalbetrieb gestört werden würde. (Keine Bedienung eines Startaufrufs während des Ablaufs der Diagnostik-Routine möglich!) Daher muß der Rechner das Starten dieser On-Line Diagnostik-Routinen über den Befehl "Diagnostik-Programm Start" übernehmen (siehe 11.1.5). Nach Ablauf der Routine übernimmt der Rechner direkt die Testergebnisse. Die Länge dieser Routinen ist auf 128 Mikrobefehlsworte begrenzt. Jeweils nur eine Routine wird in einen MIP-Speicherbereich des AW's geladen (MIP-Speicheradresse 1920 - 2047), der vom BMP mit der Führung der Fehlerstatistik (siehe Abschn. 11.1.2) belegt ist. Das BMP läuft daher nach dem Laden einer Diagnostik-Routine nur noch ohne Fehlerstatistik ab. Nach Abschluß des Wartungsbetriebs muß der Fehlerstatistik-Teil des BMP wieder über den Rechner geladen werden.

12.3.
In-Line Testbetrieb

Im In-Line Testbetrieb muß der Betriebsartenschalter des inneren Bedienfeldes SB auf Stellung "In-Line" stehen.

Die In-Line-Diagnostik-Routinen unterscheiden sich durch die On-Line-Diagnostik-Routinen nur durch die Ablaufdauer (≤ 100 ms), den Start der Routine und die Anzeige der Testergebnisse.

Die Ablaufdauer einer Routine < 100 ms erfordert keine Rechnerkoordination, da ein Startaufruf erst in einem Zeitraum bis zu 100 ms quittiert werden muß.

Die Diagnostik-Routinen können daher automatisch - AW intern - gestartet werden.

In der Nullschleife (Untätigkeitsschleife) im BMP wird laufend eine Bedingung "In-Line" abgefragt. Steht keine Startmeldung vom Rechner an und ist "In-Line" erfüllt (siehe unten), so wird die In-Line-Diagnostik-Routine automatisch gestartet und einmal durchlaufen. Anschließend erfolgt ein Rücksprung in die Nullschleife, d.h. die Rechnerpausen werden für Testbetrieb ausgenutzt. (Aufteilung des AW's in Normalbetrieb und in Rechnerpausen im Testbetrieb.)

Fehler bzw. Testergebnisse werden codiert in einem 8-Bit-Zeichen im LA-Anzeigefeld des inneren Bedienteiles angezeigt.

"In-Line"-Bedingung erfüllt bei folgenden Schalterstellungen des inneren Bedienteiles:

- SB-Schalter auf "In-Line"
- STMA-Schalter auf "Rundlauf" oder auf "Fehler" (im letzteren Fall muß Fehlercode in LA-Register = 0 sein).

Anmerkung:

Details über vorhandene Diagnostik-Routinen für alle Betriebsarten Off-Line, On-Line und In-Line-Testbetrieb sind einem separaten Wartungshandbuch zu entnehmen.

Zylinderadresse CC

Gibt die Lage aller Spuren an, die mit derselben Position der Schreib/Leseköpfe erreichbar sind.

Kopfadresse HH

Gibt die Speicherfläche an, welcher der entsprechende Kopf zugeordnet ist.

Sektoradresse S

Lage/Nummer des Sektors in der jeweiligen Spur.

ECC-Prüfzeichen (Error Correction Code)

7 Bytes am Ende eines jeden Spurabschnitts zur Erkennung und Korrektur einer bis zu 11 Bit langen Fehlstelle.

Markierungsbyte F (Flagbyte)

Dieses Byte dient zu Steuer- und Prüfzwecken sowie zur Kennzeichnung von Defektspuren und Ausweichspuren.

Physikalische Adresse (PA)

Die Physikalische Adresse gibt immer den geometrischen Ort an, auf dem sie steht.

Winkel

Enstpricht Winkelimpuls.
128 Winkelimpulse pro Umdrehung.
Feste physikalische Einteilung zur Positionsbestimmung von Sektoren auf dem Plattenstapel

Originalspurbereich:

Alle Spuren im Cylinderbereich 0-403 bei 100 MByte und
0-807 bei 200 MByte

Ausweichspurbereich:

Alle Spuren im Cylinderbereich 404-410 bei 100 MByte und
808-823 bei 200 MByte
Diese Spuren sind als Ausweichspuren für defekte Spuren im Originalbereich zuweisbar.

13.2. Die Kapazität kann nach folgenden Formeln berechnet werden:

Spurverschnitt in Bytes = 13183 - Datenblöcke/Spur • (133 + Datenlänge in Bytes)

Datenblöcke/Spur = entier [1] $\left(\frac{13183}{133 + \text{Datenlänge in Bytes}}\right)$

Datenblöcke/Zylinder = Datenblöcke/Spur • 19 [2]

Datenblöcke/Stapel = Datenblöcke/Zylinder • 404 [3] (808) [3]

Stapelkapazität in Worten = Datenlänge in Worten • Datenblöcke/Stapel

Stapelausnutzung in Prozent =

$$\frac{\text{Stapelkapazität bei der aktuellen Datenlänge}}{\text{Stapelkapazität bei 1 Datenblock/Spur}} \% =$$

$$\frac{\text{Datenlänge in Bytes} \cdot \text{Datenblöcke/Spur}}{13050} \%$$

1) entier: größte ganze Zahl enthalten in ....

2) Anzahl der nutzbaren Plattenflächen = 19

3) Anzahl der Originalspuren = 404 (808)