RECHENZENTRUM TH MÜNCHEN
ARBEITSGRUPPE BETRIEBSSYSTEME

INTERNSCHRIFT Nr.46

THEMA

Der TEBSY-Abwickler

VERFASSER

Zagler

DATUM

3.8.197o

FORM DER ABFASSUNG

- Entwurf
- Ausarbeitung
- X Endform

SACHLICHE VERBINDLICHKEIT

- Allgemeine Information
  Diskussionsgrundlage
- Erarbeiteter Vorschlag
- X Verbindliche Mitteilung
- Veraltet

ÄNDERUNGSZUSTAND

BEZUG AUF INTERNSCHRIFTEN

ANDERWEITIGE LITERATUR

Der TEBSY-Abwickler

(TEBSY = TEste BetriebsSYstem)

Inhaltsübersicht

./. 2

0. Vorbemerkung

Mit dem TEBSY-Abwickler wird eine Testhilfe für Betriebssystem-Programme geschaffen. Naturgemäß interessiert man sich bei einer Testhilfe kaum für die Konstruktionsbeschreibung und die Funktionsweise, sondern vornehmlich für die Leistungen und Steuerungsmöglichkeiten, also für die Bedienungsanleitung. Diesem Umstand wurde dadurch Rechnung getragen, daß die zur Bedienungsanleitung gehörenden Teile der Beschreibung in Abschnitt II nochmals kurz zusammengestellt werden. Für künftige Anwender des TEBSY-Abwicklers reicht es damit (und es wird empfohlen und darum gebeten), Abschnitt II zu lesen. Zu kurios anmutende Nebenbedingungen und Einschränkungen aus Abschnitt II findet man in Abschnitt I, der Beschreibung des Konstruktionskonzepts und der Funktionsweise, die erklärenden Motivierungen.

Abschnitt I: Konstruktion und Funktionsweise

1. Einführung und Problemstellung

Es ist (zunächst) eine Möglichkeit zu schaffen, für das geplante TR440-Betriebssystem BSM Testläufe auf dem TR440 im Rahmen des vom Hersteller gelieferten Betriebssystems (BS3) durchzuführen. Es ist dazu ein Programm zu erstellen, das den TR440 simuliert und auf dem simulierten Rechner den Lauf des autonomen Programms "BSM (mitsamt dessen Benutzerprozessen)" betreibt.

Die Problemstellung wird bewußt nicht derart erweitert, für jedes autonome TR440-Programm Testmöglichkeiten zu schaffen. Vielmehr werden zur Erreichung einer effektiven Simulation an die testbaren autonomen Programme einschränkende Bedingungen gestellt, die vom BSM ohne besondere Schwierigkeit erfüllt werden können (es genügt, diese Bedingungen bei der Codierung zu berücksichtigen).

./. 3

Die Simulation eines Rechners auf sich selbst läßt sich in vielen Fällen recht einfach vollziehen. Sofern die zu simulierenden Register gerade durch die entsprechenden realen Register verwirklicht werden, läßt sich die Simulation eines Befehls durch reale Anwendung desselben Befehls auf den realen Rechner bewerkstelligen (triviale Simulation). Diese triviale Simulation hat neben der Einfachheit der Durchführung auch noch den Vorteil, einen Zeitverlust zu vermeiden.

Es wird ausdrücklich als zur Problemstellung gehörig bezeichnet, daß aus Effektivitätsgründen die Gelegenheiten zur trivialen Simulation möglichst umfassend zu nutzen sind. Demgegenüber ist etwa das Schaffen und Anbieten von Testhilfen vorerst zweitrangig. Damit ist die Grundkonzeption eines jeglichen Lösungsansatzes weitgehend festgelegt.

Zusammenfassend und in der für weitere Anwendbarkeit notwendigen Weise verallgemeinert, heißt damit die Problemstellung: Es ist ein Programm zu schaffen, das den TR440 auf dem TR440 zum Zwecke des Testens von Betriebssystemen simuliert (TEBSY-Programm). Dabei sind die Möglichkeiten zu einer trivialen Simulation voll auszuschöpfen, auch wenn dies zu gewissen einschränkenden Bedingungen an die testbaren Betriebssysteme (Testsysteme) führt. Die Bedingungen, denen die Testsysteme zu genügen haben, werden deutlich herauszustellen sein. Das TEBSY-Programm läuft im Rahmen eines übergeordneten Betriebssystems (Rahmensystem). Es ist auf gewisse, mitunter sehr spezielle Versorgungs- und Dienstleistungen des Rahmensystems angewiesen, die ebenfalls angegeben werden.

Falls die Gefahr einer Fehldeutung durch Verwechslung von realen und simuliertem Rechner besteht, bedienen wir uns folgender Redeweise: Begriffe und Vorgänge, die sich auf den simulierten Rechner beziehen, werden durch Voransetzen der Vorsilbe "pseudo" genauer gekennzeichnet.

./. 4

2. Grundkonzeption

Die Eingliederung von TEBSY in ein Rahmensystem erfordert, daß es im Normal- oder Abwicklermodus läuft (Hypothese). Die triviale Simulation als häufigste und einfachste Simulationsart (häufig vielleicht nicht im Sinne des darauf entfallenden Zeitaufwandes, aber hinsichtlich der Anzahl der davon betroffenen Befehle) geschieht im Normalmodus (Hypothese).

Zur Verwirklichung der Grundkonzeption wird folgender Grundsatz aufgestellt: Die Simulation der Ausführung eines Befehls wird durch ungeänderte Anwendung des betreffenden Befehls auf den realen Rechner ausgelöst (angestoßen). Danach ergibt sich für die Forderung, möglichst weitgehend von der trivialen Simulation Gebrauch zu machen, folgende Neuformulierung: Die gesamte Simulation ist so einzurichten, daß durch die obige Art der Auslösung möglichst weitgehend auch der endgültige und vollständige Vollzug der Simulation der Befehlsausführung stattfindet.

Damit sind nur solche Befehle für eine triviale Simulation geeignet, die im Normalmodus zulässig sind. Die im Normalmodus unzulässigen Befehle (wie beispielsweise LEI, VMO, VPU) werden bei der Simulation im Normalmodus gegeben und führen zu einer Makro-Unterbrechung. Diese Makro-Unterbrechungen werden vom Rahmensystem mit ausreichender Information versehen an das TEBSY-Programm zurückgeleitet, womit von diesem anschließend die echte (nichttriviale) Simulation vorgenommen werden kann.

Eine besondere Behandlung erfordern diejenigen "kritischen" Befehle, die zwar im Normalmodus zulässig sind, bei denen aber die ungeänderte Anwendung auf den realen Rechner den Vollzug der Simulation nicht erbringt. Hier muß vom den eingangs erwähnten Grundsatz abgewichen werden und ein anderer Befehl auf den Rechner angewendet werden. Die nötige echte

./. 5

Simulation läßt sich durch Verwendung eines Leercodes mit Hilfe der zugehörigen Makro-Unterbrechung erreichen (auslösen). Die Einfügung eines Leercodes anstelle eines kritischen Codes läßt sich bei der TAS-Quelle des Testsystems durch die Ersetzungstechnik leicht erreichen. An Testsysteme ergeben sich daraus folgende Einschränkungen:

Leercodes, die vom Systemtestprogramm zur Ersetzung kritischer Codes verwendet werden, dürfen im Testsystem nicht vorkommen.

Kritische Codes dürfen vom Testsystem nicht errechnet werden (etwa als Vorbereitung zu einem Befehl wie R T A). Schärfer und präziser heißt die Einschränkung: Das Testsystem darf bei kritischen Befehlen die Kenntnis des Interncodes nicht ausnutzen.

Nach dem gegenwärtigen Stand sind folgende Befehle kritisch:

BSS: Siehe Beschreibung zu W2, W3, W4 im leeren Rechner: Im Normalmodus keine Wirkung, aber zulässig.

BLEI: Eine triviale Simulation von BLEI würde erfordern, im Leitblock des Systemtest-Programms die Angaben des aktuellen Pseudoleitblocks zu halten. Wegen der Eingliederung in ein Rahmensystem ist aber der Leitblock unantastbar.

BCI, ZI: Der Stand des Indexbasisregisters muß nicht mit einem Inhalt der Indexbasis-Ablageplätze (absolut 4 bzw. Zelle 4 in einem Leitblock) korrespondieren. Bei Simulation solcher Zustände ist das Indexbasisregister echt zu simulieren. Danach ist ein elementarer Bezug auf den realen Leitblock und dessen Kacheltabelle nicht mehr möglich.

./. 6

Wie der BSS-Befehl ist auch der SSR-Befehl stark modusabhängig. Falls die SSR-Behandlung des zum TEBSY-Programm gehörigen Abwicklers die echte Simulation des SSR-Befehls durchführt, ist eine Ersetzung des SSR-Befehls durch Leercode unnötig.

Die spezielle Art der Makrobehandlung, die direkte Anwendbarkeit des SSR-Befehls, weitere gelegentlich nötige Auszeichnungen und Bevorzugungen des TEBSY-Programms durch das Rahmensystem und eine zweckmäßige Angleichung an das BSM (sobald dieses auch als Rahmensystem fungieren kann) lassen es geraten erscheinen, für das TEBSY-Programm neben dem Normalmodus auch den Abwicklermodus als realen Betriebsmodus vorzusehen.
Noch zwingender folgt dies aus der zu wählenden Adressierungsmethode (3.), durch die für alle übrigen im Normalmodus zulässigen Befehle die triviale Simulation ermöglicht wird.

Wir werden daher nachfolgend nur noch vom TEBSY-Abwickler sprechen. Er stößt den Testlauf für das Testsystem durch Übergang in den Normalmodus an. Solange der Normalmodus beibehalten bleibt, vollzieht sich die triviale Simulation. Durch besondere Vorkommnisse (SSR-Bef, Makros, Alarme, ...) gelangt man schließlich wieder in den ST-Abwickler, in dem die echte Simulation durchgeführt wird.

3. Adressierung

Die triviale Simulation vollzieht sich im realen Normalmodus. Dabei sind die Bereichsangaben Δ^{o}_{1}, Δ^{o}_{2} pseudomodusabhängig eingestellt und verweisen auf einen geeignet vorbereiteten Abschnitt der TEBSY-Kacheltabelle.

3.1 Pseudo-Systemmodus

Die eingestellten (einzustellenden) Bereichsangaben Δ^{o}_{1}, Δ^{o}_{2} verweisen auf einen Abschnitt, den wir Absolutadressentabelle nennen. Die Absolutadressierung des Pseudosystemmodus wird durch die Relativadressierung über den Leitblock mit Hilfe der Absolutadressentabelle simuliert.

./. 7

3.2 Pseudo-Normal/Abwickler-Modus

Die eingestellten (einzustellenden) Bereichsangaben Δ_1^o, Δ_2^o verweisen auf einen Abschnitt der Kacheltabelle, den wir Relativadressentabelle nennen und mit deren Hilfe sich die Simulation der Relativadressierung im Pseudo-Normal/Abwickler-Modus vollzieht.

3.3 Pseudo-Spezialmodus

Der Spezialmodus mit seinen zwei Adressierungsarten (BF enthält Absolutadressen (der Befehle), Speicheradressen (Daten) relativ über Kacheltabelle) ist schwieriger trivial zu simulieren. Die Lösung benutzt die besondere Eigenschaft der "normalen" Sprungbefehle, nur die letzten 16 Bits von BF neu zu besetzen und die ersten 8 Bits nicht zu verändern. Der gesamte Inhalt von BF wird nur bei den "Großseitensprüngen" mit Hilfe von SE, SFBE, SUE und mit MABI oder MU modifizierten normalen Sprungbefehlen neu besetzt.

Verzichtet man nunmehr im Pseudospezialmodus auf Großseitensprünge der obigen Art und befindet sich in den ersten 8 Bits von BF erst einmal die passende Großseitennummer, so werden nur noch Befehle aus dieser eingestellten Großseite angesprochen, unabhängig vom eingestellten Δ_1^o-Wert.

Zur Relativadressierung der Daten wird somit Δ_1^o wie im Pseudonormalmodus auf den Anfang der Relativadressentabelle eingestellt. Der einzustellende Δ_2^o-Wert muß aber über das Ende der Relativadressentabelle hinausgehend noch den (pseudo-absoluten) Zugriff auf die Großseite für die Befehle gestatten, was über den hierzu nötigen Teil der Absolutadressentabelle geschieht. Insbesondere folgt daraus, daß im TEBSY-Leitblock die Relativadressentabelle vor der Absolutadressentabelle angeordnet sein muß.

./. 8

Bei dieser Lösung wird also Pseudobefehlszähler mit dem realen Befehlszähler so simuliert, daß letzterer hinsichtlich der eingestellten Großseite geeignet falsch geht.

Es ist zu betonen, daß damit nicht alle Großseitensprünge im Pseudospezialmodus fehlgehen, sondern nur solche, die trivial simuliert werden. Eine Ausweichmöglichkeit mit nur geringfügig unterschiedlicher Handhabung bietet der VMO O 1; dieser Großseitensprung wird durch den TEBSY-Abwickler echt und richtig simuliert. Analoges gilt für die Ablage des BF bei Unterbrechungen; auch dies wird durch echte Simulation erledigt, wobei der während der trivialen Simulation verfälscht eingestellte BF vorher berichtigt wird.

Der falsch gehende BF führt bei den Befehlen der Unterprogramm-Technik zu kuriosen und recht nützlichen Konsequenzen. Bei den Ansprungbefehlen SU bzw. SFB wird der verfälschte BF-Stand reserviert. Falls dieser zu einem Großseiten-Rückkehrsprung verwendet wird, ist alles wieder in Ordnung. Die den Ansprungbefehlen in natürlicher Weise zugeordneten Rückkehrbefehlen, nämlich

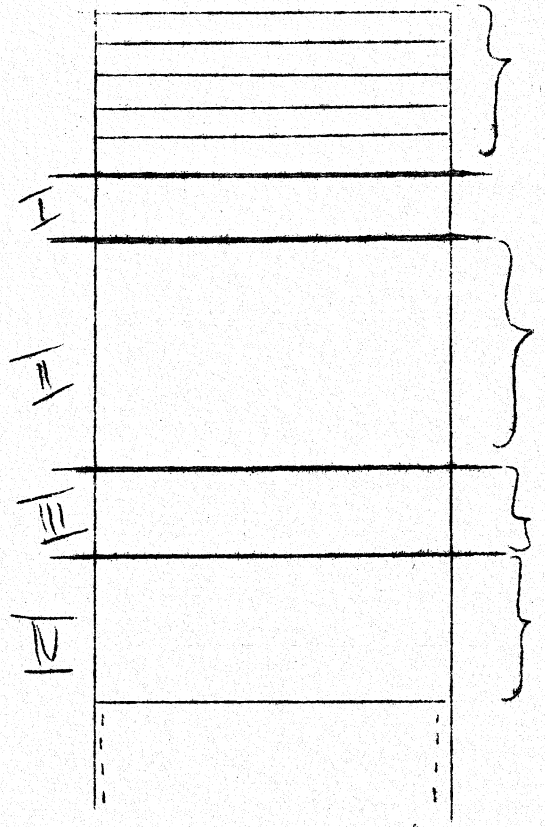
SU UP ⟶ MU S O,
SFB UP ⟶ TBC N, ..., SE N,
oder MABI S O,

sind Großseitensprünge und funktionieren somit richtig. (Die Ansprungbefehle SUE und SFBE sind als Großseitensprünge nach wie vor unzulässig; nur für UP-Rücksprünge sind Großseitensprünge passend.)

3.4 Einteilung des TEBSY-Leitblocks

Die eingetragenen Zahlanangaben sind als realistisch gewählte, änderungsfähige Beispiele zu verstehen.

./. 9



5 Teilleitblöcke (à 16 gw)
für Hardware- und BS3-Eintragungen

1 Teilleitblock TEBSY-Eigenbereich
zur Durchführung der echten Simulation

12 Teilleitblöcke für Relativadressen-
tabelle (entspricht 768K Adreßraum)

7 Teilleitblöcke Lücke

12 Teilleitblöcke für Absolutadressen-
tabelle (entspricht 768K Kernspeicher)

Einstellung der $\triangle$ -Werte (überstrichener Bereich):

a) Pseudo-Normal/Abwickler-Modus II

b) Pseudo-Systemmodus IV

c) Pseudo-Spezialmodus II, III (ganz)
und IV (evtl. nur teilweise)

d) Echte Simulation: I, II, III, IV

Im Pseudo-Spezialmodus ist der Speicherschutz aufgeweicht:
Zu große Relativadressen für Daten führen in die Absolutadressen-
tabelle und werden nicht als unzulässig erkannt. Hier kann man
teilweise eine Verbesserung erreichen, indem man zwischen I
und II eine Lücke von etlichen Teilleitblöcken (die im Zahlen-
beispiel gewählte Anzahl 7 ist wegen d) maximal) einschiebt. Es
werden dann wenigstens diejenigen unzulässigen Relativadressen
angezeigt, die in die Lücke führen.

./. 1o

Die für die Relativadressentabelle vorzusehende maximale Länge (im Beispiel 12 Teilleitblöcke) zieht eine weitere Einschränkung für Testsysteme nach sich: Die Länge des durch die Pseudo-Delta-Register angesteuerten Abschnitt eines Pseudoleitblocks darf die maximale Länge der Relativadressentabelle nicht überschreiten.

4. Aufbereitung der Kacheltabelle

Alle Kacheln, die der TEBSY-Abwickler zum Zwecke der Kernspeichersimulation vom Rahmensystem anfordert und erhält, werden zunächst in die Absolutadressentabelle eingetragen, die die Abbildung Pseudokachel ⟼ Kachel vermittelt. Dabei ist überall das Schreibschutzbit zu löschen entsprechend dem ungehinderten Zugriff im Pseudo-Systemmodus.

Die in den Pseudo-Leitblöcken des Testsystems befindlichen Pseudo-Kacheltabellen vermitteln die Abbildung Pseudoseite ⟼ Pseudokachel. Durch Zusammensetzen beider Abbildungen ist daraus die Relativadressentabelle, die die Abbildung Pseudoseite ⟼ Kachel vermittelt, aufzubauen.

Pseudoseite —(Pseudo-kacheltabelle)→ Pseudokachel —(Absolut-adressentabelle)→ Kachel

Pseudoseite —(Relativadressentabelle)→ Kachel

Der in der Pseudokacheltabelle enthaltene Speicherschutz ist dabei folgendermaßen zu berücksichtigen:

a) Für jede Pseudoseite ist das Schreibschutzbit in der Relativadressentabelle ebenso zu setzen wie in der Pseudokacheltabelle.

./. 11

b) Jeder Pseudoseite, der in der Pseudokacheltabelle die Pseudokachel 0 zugeordnet ist, ist auch in der Relativadressentabelle die Kachel 0 zuzuordnen (Kennzeichen für "nicht eingetragen"). Will man das Schema der Hintereinanderschaltung der zwei Abbildungen beibehalten, was sich empfiehlt, so ist die durch die Absolutadressentabelle vermittelte Abbildung durch die eine Festsetzung Pseudokachel 0 ⟶ Kachel 0 abzuändern.

Eine Änderung der Pseudokacheltabelle zieht eine Änderung der Relativadressentabelle nach sich, die vom TEBSY-Abwickler zu besorgen (anzustoßen) ist; er muß also von der Notwendigkeit einer solchen Änderung erfahren. Eine Änderung der Pseudokacheltabelle ist daher nur zulässig, wenn die Relativadressentabelle bei der Simulation nicht benutzt wird, also im Pseudo-Systemmodus, oder wenn die Änderung mit Hilfe eines privilegierten Befehls geschieht, der zur echten Simulation in den TEBSY-Abwickler führt. Daraus ergibt sich als Bedingung an Testsysteme: Abspeicherungen im aktuellen (durch die Pseudo-BLZ1/BLZ2-Register bezeichneten) Abschnitt einer Pseudokacheltabelle dürfen in einem Nicht-Systemmodus nicht mit Befehlen ausgeführt werden, die trivial simuliert werden.

Solche unzulässigen Abspeicherungen können mit trivial simulierten Befehlen nur dann eintreten, wenn der Pseudoleitblock in den Adressenraum eines Pseudoprozesses schreibungeschützt eingebettet ist. Aus Sicherheitsgründen wird aber ein Betriebssystem den Prozessen die eigenständige Bearbeitung des Leitblocks ohnehin nicht gestatten, indem es solche Dienstleistungen nicht anbietet. Würde ein Testsystem diesbezüglich einem (besonders zuverlässigen) Prozeß eine Ausnahmegenehmigung erteilen und dieser selbständige Änderungen des aktuellen Pseudokacheltabellenabschnitts vornehmen, so kann dessen Tätigkeit im vorgelegten Konzept nicht simuliert werden.

./. 12

Für den TEBSY-Abwickler verliert bei Eintritt in den Pseudosystemmodus die Relativadressentabelle ihre Gültigkeit. Vor (bei) Verlassen des Pseudosystemmodus ist die Relativadressentabelle neu zu erstellen, übrigens auch dann, wenn sich die aktuelle Pseudokacheltabelle nicht geändert hat, da sie sich unbemerkt hätte ändern können. Sonstige Änderungen der Relativadressentabelle rühren von privilegierten Befehlen her und sind bei der echten Simulation zu erledigen.

Erstellung und Änderung der Relativadressentabelle erfordert offenbar eine sehr spezielle Unterstützung des TEBSY-Abwicklers durch das Rahmensystem. Ob und wie dies einmal das BSM als Rahmensystem bewerkstelligen kann, wäre wohl interessant zu überlegen.

Die möglicherweise häufig nötige Neuerstellung der Relativadressentabelle bei Verlassen des Pseudosystemmodus ist der Preis, der für die zeitverlustfreie, triviale Simulation zu zahlen ist. Bei sparsamster Programmierung unter passender Verwendung des US-Befehls und ohne Berücksichtigung der Rahmenorganisation, benötigt die Aufbereitung eines Teilleitblocks der Relativadressentabelle etwa 0,2 msec.

5. Simultanverarbeitung und Unterbrechungen der Rechnerkernsimulation

Die Simultanverarbeitung durch mehrere Rechnerkerne und durch das EA-Werk wird im TEBSY-Abwickler, dem jeweils nur 1 Rechnerkern zur Verfügung steht, durch Zeitverschränkung simuliert. Das die Zeitverschränkung bewerkstelligende Programmstück des TEBSY-Abwicklers nennen wir die Abwickler-Vergabe.

./. 13

Zur Steuerung der Zeitverschränkung wird je Rechnerkern und je EA-Kanal eine Größe Gerätezeit geführt, die angibt, bis zu welchem gedachten Zeitpunkt seit Testlaufbeginn die Tätigkeiten der zugehörigen Maschinenkomponente schon simuliert worden sind. Demgemäß ist die Buchführung und Fortschreibung für die Gerätezeit einzurichten. Beispielsweise wird beim Anstoß eines ruhenden Kanalwerks durch Y-Befehl dessen Gerätezeit auf die Gerätezeit des anstoßenden Rechnerkerns (nachdem diese ggf. aktualisiert wurde) gesetzt.

Um die Zeitverschränkung von Pseudo-Vorgängen zu bewerkstelligen, ist eine gelegentliche Unterbrechung dieser Pseudo-Vorgänge erforderlich. Zugleich ist dabei die Simulation der Kommunikation zwischen den Maschinenkomponenten vorzubereiten, die sich folgender Hilfsmittel bedient:

Eingriffe:	Kanalwerk ⟶ Rechnerkern	
Y-Befehl:	Rechnerkern ⟶ Kanalwerk	
	Rechnerkern ⟶ Rechnerkern	(RK-Alarm)

Im folgenden werden wir nur die Unterbrechung der Rechnerkernsimulation besprechen.

Zur Rechnerkernunterbrechung zwecks Zeitverschränkung können beliebige Vorkommnisse im Abwickler, beispielsweise alle Eintritte, verwendet werden. Zur halbwegs realistischen Simulation des zeitlichen Zusammenspiels der Maschinenkomponenten reicht es aber mitunter nicht aus, allein die vom Lauf des Testsystems hervorgerufenen Eintritte in den Abwickler zur Zeitverschränkung zu benutzen. Es könnte über zu lange Zeitabschnitte hinweg die triviale Simulation anhalten, während der am Pseudorechner gleichsam ungewollt Eingriffssperre und eine Alarmsperre für echt zu simulierende Alarme (siehe 7.) herrscht. Daher ist vom Rahmensystem eine Möglichkeit zur Unterbrechung der trivialen Simulation, also des Laufs im realen Normalmodus, zu fordern.

./. 14

Hierzu wäre ein Weckdienst passend. Behelfsweise kann er durch ein genügend (aber nicht zu) häufiges unmotiviertes Wecken, etwa in gleichen Zeitabständen, ersetzt werden. In diesem Falle ist bei der Kommunikation mit gewissen Zeitdifferenzen zu rechnen. Ebenso wird der Pseudo-Wecker eine entsprechend grob arbeitende Einrichtung des Pseudorechners.

Zur Wahl eines geeigneten Unterbrechungszustandes, mit dem ein Pseudorechner bei der Zeitverschränkung ruht, beachte man die möglichen Kommunikationsmeldungen, die vom Pseudorechner bei Wiederaufnahme zu berücksichtigen sind: Eingriffe und Rechnerkernalarme. Da das Vorliegen solcher Meldungen am Ende der Abrufphase abgefragt wird, empfiehlt es sich, auf diesen Moment auch den Unterbrechungszustand zu legen. Ohne unnötige Komplikationen können dann echt zu simulierende Alarme und Eingriffe durch Setzen der entsprechenden Pseudo-Flipflops erledigt werden. Zudem entspricht dies der durch den Wecker erzeugten Abbruchsituation.

Neben den vom Wecker erzeugten Unterbrechungen werden auch die bei der trivialen Simulation auftretenden Alarme als Zeitverschränkungsunterbrechung ausgenutzt. Dadurch wird ein gleichzeitiges Auftreten von echt und trivial simulierten Alarmen ermöglicht.

./. 15

Trennblatt zu Abschnitt II

Sogar die eifrigen Leser von Abschnitt I werden gebeten, sich noch Abschnitt II zu Gemüte zu führen, da die dort zusammengefaßten Benutzungshinweise auch solche aus den in Vorbereitung befindlichen Paragraphen

6. Makro- und SSR-Befehle
7. Simulation von Alarmen
8. Bemerkungen zu den echt simulierten Befehlen

von Abschnitt I enthalten.

./. 16

Abschnitt II: Hinweise für die Benutzung

Der TEBSY-Abwickler simuliert auf dem TR440 im realen Abwickler- und Normalmodus eine im wesentlichen vollständige TR440-Hardware. Mit Rücksicht auf die Effektivität unterliegt die simulierte Hardware gewissen Einschränkungen und erfordert gelegentlich eine Anpassung der testbaren Programme.

Diese Besonderheiten werden nachfolgend in einem ersten, unvollständigen Entwurf zusammengestellt:

1. Ersatzmakros: Anstelle der Befehle BCI, BLEI, BSS und ZI ist *(BCI), *(BLEI), *(BSS) und *(ZI) zu schreiben. Mit Hilfe dieser TAS-Ersetzungen werden ersetzende Makrocodes eingefügt, gegenwärtig 'E8', '9E', 'FC' und 'E9'. Für den Lauf auf der realen Maschine kann damit auch leicht zu den Originalcodes zurückgekehrt werden.

   Die ersetzenden Makrocodes sind nicht mehr als Makros zu verwenden.

   Für Testläufe auf dem TR4 mit dem zugehörigen TR440-Simulator (unausweichlich bis Ende September 7o) sind solche Ersetzungen auch für die Befehle SW, VMO, VPU und VSS vorzunehmen.

2. Im Spezialmodus geht das Befehlsfolgeregister BF in den ersten 8 Bits falsch außer bei Unterbrechungen, privilegierten Befehlen und Befehlen mit Ersatzmakros. Dies ist bei den Unterprogrammansprüngen SU und SFB zu beachten. Die reservierte Rückkehradresse ist nur für den Unterprogrammrücksprung (MU S O, MABI S O) passend, und das auch nur dann, wenn bei der Rückkehr wiederum Spezialmodus vorliegt.

   Im Spezialmodus werden nichtprivilegierte Großseitensprünge (das sind solche, die beim BF auch die ersten 8 Bits neu

./. 17

besetzen, beispielsweise SE, SUE, SFBE) falsch ausgeführt und sind daher unzulässig mit Ausnahme der Unterprogramm-rückkehrsprünge, wo Großseitensprünge vorgeschrieben sind. Notfalls kann auf den privilegierten Großseitensprung VMO 0 1 ausgewichen werden.

3. Generell darf das BLZ2-Register nur mit Werten ≤ 11 geladen werden (sonst erfolgt Hinweis und Einsetzen des Maximal-wertes 11).

4. Im Spezialmodus sollte die aktuelle (durch die Register BL, BLZ1 und BLZ2 eingestellte) Kacheltabelle nicht mit nicht-privilegierten Befehlen verändert werden. Die Abspeicherung wird im simulierten Kernspeicher zwar durchgeführt, eine Änderung der Seitenzuordnung bei der Adressierung aber nicht wahrgenommen.

   Eine solche Änderung durch nichtprivilegierte Befehle kann nur dann zustande kommen, wenn der Leitblock in den Adressenraum schreibungeschützt eingebettet ist.

5. Die Simulation von zeitbedingten Abläufen geschieht mit einem mittleren Zeitfehler von T/2, wobei T noch festzu-setzen und voraussichtlich zwischen 1 und 100 msec liegt. Davon betroffen sind Weckeralarme, Rechnerkernalarme, Eingriffe und EA-Kanalanstöße durch Y-Befehl.

6. Die Steuerbits 35 (Wartungsmodus) und 34 (Modus 16) gibt es nicht.

7. Beim LEI-Befehl sind als Zweitcode nur solche Codes zugelassen, die in der Befehlsliste mit Adressteilen m, n oder z versehen sind. Für die zugelassenen Zweitcodes BCI, ZI und VSS mit Er-satzmakros ist als Zweitcode *(BCIBCI), *(ZIZI) und *(VSSVSS) einzusetzen.