

Simulationssprache GASP II für
Prozeßrechner Siemens 305

A. Schüring

Technische Hochschule Aachen,
Laboratorium für Werkzeugmaschinen und
Betriebslehre

Simulationssprache GASP II für Prozeßrechner SIEMENS 305

Werkzeugmaschinenlabor

Lehrstuhl M. Weck, TH Aachen

Dipl.-Ing. A. Schüring

1. Einführung

Zu Untersuchung diskontinuierlicher Vorgänge eignet sich die Technik der zeitdiskreten Simulation auf einem Digitalrechner. Dazu wird ein programmiertes mathematisches Modell der realen Vorgänge erstellt, mit dem das dynamische Verhalten des nachgebildeten Systems studiert werden kann. Das erforderliche Modell wird gewöhnlich in einer allgemeinen problemorientierten Sprache oder einer höheren Simulationssprache abgefaßt.

Im Bild (Bild 1) sind einige Simulationsmodelle (software packages) aufgeführt. Die Modelle GASP, GPSS, SIMULA und SIMSCRIPT eignen sich wegen ihrer diskreten Zeitstruktur für die Nachbildung diskontinuierlicher Systeme. Während GASP aus einer Reihe von FORTRAN-Unterprogrammen besteht, werden die anderen drei in höheren Simulationssprachen programmiert. Wegen der simulationsspezifischen Sprachelemente gibt es für sie eigene Compiler. Diese Compiler erfordern bei großem Komfort auch große Rechenanlagen, sind aber nicht für alle Anlagen verfügbar. Außerdem sind die Modelle in ihrem Aufbau gegenüber Änderungen ziemlich unflexibel und verlangen eine mehr oder weniger aufwendige Einarbeitung durch den Anwender. Das Simulationsmodell GASP dagegen kann von einem Standard-FORTRAN-Compiler übersetzt werden; es läßt sich individuell verändern und ist auch auf Anlagen von 16 K-Worten an aufwärts einzusetzen.

Das Simulationspaket GASP II [1] sollte auf dem Prozeßrechner implementiert werden. Es zeigte sich jedoch, daß das gesamte Paket einschließlich problemspezifischem Teil nicht in dem verfügbaren Zentralspeicherbereich untergebracht werden konnte. Daher galt es, das gesamte Paket in lauffähige Segmente aufzuteilen.

2. Aufbau und Funktion von GASP II

GASP II beinhaltet eine Reihe von FORTRAN-Unterprogrammen, die dem Anwender die bei einer Simulation immer wiederkehrenden Programmierarbeiten weitgehend abnehmen. Das sind vornehmlich folgende Aufgaben (Bild 2):

- zeitliche Fortschaltung der Simulationszeit
- Führen des Ereigniskalenders
- Ein- und Ausreihen von Einheiten in und aus Wartebereichen
- statistische Erfassung und Auswertung

Die einzelnen Funktionen der Unterprogramme sind im Bild wiedergegeben. Die meisten Programme sind Bestandteile von GASP, die restlichen sind vom Anwender zu schreiben. Dazu gehört auch das Hauptprogramm MAIN, in dem die "Nicht-GASP-Größen" initialisiert werden. Problemspezifisch sind die Ereignisunterprogramme, welche die Ereignisse im Simulationsmodell bearbeiten. Die Gesamtspeicherlänge des Simulationspaketes hängt einerseits von der Anzahl der im Modell notwendigen Einheiten und Ereignisse ab, andererseits von der Länge der einzelnen Ereignisunterprogramme. Im folgenden wird ein kurzer Überblick über die Funktionen der GASP-Programme gegeben:

Das Unterprogramm (UP) GASP steuert den gesamten Ablauf der Simulation. Im folgenden UP DATAN werden die GASP-Variablen und Konstanten eingelesen und zu Beginn vom UP SET in ein zentrales Simulationsfeld eingetragen. Dort sind auch die Ereignisse und Einheiten mit ihren Attributen gespeichert. Das Simulationsfeld steht nicht im COMMON-Bereich, es wird im Hauptprogramm dimensioniert und ist daher von Lauf zu Lauf in seinen Abmessungen leicht zu ändern. Mit Hilfe des UP MONTR kann dieses Feld zur Kontrolle des Simulationslaufes zwischenzeitlich ausgedruckt werden.

GASP aktiviert über das UP EVNTS die vom Anwender geschriebenen problemspezifischen Ereignisprogramme. Dazu bedient es sich eines Ereignis-Kalenders, der in dem zentralen Simulationsfeld mit Hilfe einer Pseudo-Simulationsuhr bearbeitet wird. Die bearbeiteten Ereignisse stoßen durch Kalendereintragen neue innere Ereignisse an. Das Ein- und Austragen von Einheiten in und aus Wartebereichen wird von den Unterprogrammen FILEM und RMOVE ausgeführt. Die Programme COLCT, TMST, HISTO sammeln die während der Simulation anfallenden Daten und verdichten sie zu Statistiken, die von UP SUMRY ausgedruckt werden.

Zur Nachbildung stochastischer Vorgänge sind programmierte Standard-Zufallszahlengeneratoren vorhanden, andere können vom Anwender hinzugefügt werden.

Zusammenfassend kann festgestellt werden, daß die GASP-Systemprogramme unverändert übernommen werden können und darüber hinaus spezielle FORTRAN-Anwenderprogramme mühelos zu integrieren sind. Der Ablauf einer GASP-Simulation (Bild 3) ist durch die Hauptfunktionen "Initialisierung, Ereignis-Steuerung, Überwachung und Auswertung" gekennzeichnet.

Die Kontrolle und Fehlersuche im Simulationsprogramm vereinfachen die Funktionen "Programm-Trace" und "Fehler-Report". Zur Aufteilung des gesamten Paketes in Segmente sind die einzelnen Programme auf diese Segmente so zu verteilen, daß die Simulationszeit nicht wesentlich verlängert wird.

3. Segmentierung von GASP

Während des Simulationslaufes wechselt die Bearbeitung zwischen den einzelnen Unterprogrammen. Einige werden selten aufgerufen, andere aber sehr häufig. Die Verteilung dieser Unterprogramme ist nun so vorzunehmen, daß die Zahl der Segmentwechsel in Grenzen bleibt. Da der interne Ablauf aber recht undurchsichtig ist, wurde eine experimentelle Lösung gefunden, um die wechselseitigen Beziehungen, d. h. SUBROUTINE-CALLS erfassen zu können. Auf einem Großrechner wurde ein Simulationsmodell unsegmentiert bearbeitet und anschließend die Summe aller gegenseitigen CALL- und RETURN-Aufrufe ausgedruckt (Bild 4).

Zur Vereinfachung sind die gefundenen Zusammenhänge hier in drei Häufigkeitsklassen dargestellt [2]. Danach tritt ein Wechsel zwischen GASP und dem Austrageprogramm RMOVE sehr häufig, zwischen GASP und DATAN jedoch nur selten auf. Daraus ist zu folgern, daß die erstgenannten möglichst im selben Segment liegen sollten. Anderenfalls ergeben sich viele Segmentwechsel und damit auch lange Simulationszeiten.

Wählt man ein anderes Simulationsbeispiel, so ändern sich auch die Wechsel-Häufigkeiten. Das gilt aber weniger für die GASP-Systemprogramme als für die vom Anwender geschriebenen. Da deren Verknüpfungen ihm aber bekannt sind, kann er eine günstige Segmentstruktur leicht bestimmen.

Zentralspeicher-resident (Bild 5) sind der sogenannte Segmentkopf und der COMMON-Bereich, die zu Beginn vom Externspeicher geholt werden. Um diese und die Segmentwechselzeiten möglichst klein zu halten wurden alle Teile absolut-adressiert auf dem Externspeicher (ESP) abgelegt. Sie können also ohne Bereitstell-Aufruf in den Zentralspeicher (ZSP) übertragen werden. Die Aufteilung des ZSP-Platzes gliedert sich folgendermaßen:

3,7 K-Worte wurden vom ORG belegt, der Segmentkopf war etwa 200 Worte lang. Für das COMMON-Feld wurde ein Bereich von 2,5 K freigehalten, um das Simulationsmodell noch ausdehnen zu können [3]. Für den Segment-Laufbereich blieb also noch ein Platz von 9,6 K. Im gewählten Beispiel war das größte Segment etwa 7 K lang. Das unsegmentierte Paket einschließlich der Felder benötigte in einer bescheidenen Ausbaustufe bereits 12,5 K Worte ZSP-Platz, belegte also schon den gesamten neben dem ORG verfügbaren Platz. In den Segmenten 1 und 2 werden die Vorbereitungen für die Simulation getroffen, Segment 4 beinhaltet vor allem Auswerte- und Druckprogramme. Die eigentliche Simulation wird im dritten Segment durchgeführt. Da es die meisten Programme enthalten muß, wird dort die Speichergrenze am ehesten erreicht.

Der Wechsel der Segmente wird vom ZSP-residenten Segmentkopf durchgeführt (Bild 6). Dieses Assemblerprogramm läuft in einer Schleife und prüft bei jedem Durchlauf, ob ein Segment auf ENDE gelaufen ist. Aus einer Segmentanforderungszelle entnimmt es die Nummer *k* des Folgesegments, überträgt es in den ZSP und startet es. Diese Nummer

wurde vom Segment 1 dort eingeschrieben. Außerdem wird noch eine Verzweigungsanweisung übergeben, die den Einsprung in das Segment k bestimmt.

Das Simulationspaket liegt in FORTRAN 300 vor, kann aber leicht für den FORTRAN-IV-H-Level Compiler aufbereitet werden. Für das schon erwähnte Beispiel wurde ein Vergleich der segmentierten Version mit der unsegmentierten Großrechnerversion durchgeführt. Bei nahezu gleicher Genauigkeit dauerte der Simulationslauf 10 min auf der S 305 gegenüber etwa 60 Sekunden auf einem CONTROL DATA 6400 Großrechner. Die zeitliche Dehnung des Simulationslaufes auf der S 305 wurde aber nur unwesentlich durch die Segmentierung bedingt, da nur insgesamt fünf Segmentwechsel ausgeführt wurden. Zur Zeit werden am Werkzeugmaschinenlabor der TH Aachen mit Hilfe von GASP Fertigungssysteme simuliert und analysiert. Ein Beispiel dazu wird im folgenden gegeben.

4. Beispiel zur Anwendung

Flexible Fertigungssysteme (Bild 7) bestehen aus Fertigungseinrichtungen die über Transporteinrichtungen miteinander verkettet sind. Damit ist ein automatischer rechnergesteuerter Werkstück- und Werkzeugfluß möglich. Zur Bestimmung der Auslegung und einer Vorab-Analyse eignet sich die zeitdiskrete Gesamtsimulation. Im Modell ist eine Nachbildung der Transportvorgänge und der Datenanforderungen der Maschinen an den Prozeßrechner programmiert. Durch Parametervariation in mehreren Läufen läßt ^{sich} eine geeignete Lösung erarbeiten. Als besondere Vorteile der GASP-Simulation können die Transparenz der in FORTRAN geschriebenen Modelle sowie die einfachen Möglichkeiten der Veränderung und Erweiterung durch den Anwender angesehen werden.

Literaturverzeichnis

- | | |
|--------------------------------|---|
| [1] Pritsker, A.
Kiviat, P. | Simulation with GASP II
Prentice-Hall, Inc. 1969 |
| [2] Junge, J. | Nachbildung rechnergeführter Fertigungssysteme
Diplomarbeit TH Aachen 1974 |
| [3] Reinhardt, A. | TZ für praktische Metallverarbeitung
Jg. 66 (1972), H. 2 |

Bild 1:
Auswahl einiger programmierbarer Simulationsmodelle

Modell	Erläuterung des Namens	entwickelt von	Sprach-Basis	Struktur	Zeit-Ablauf	Bemerkungen
DYNAMO	Dynamic Models	MIT (USA)	—	quasi-kontinuierlich	perioden-orientiert	Simulation von Regelsystemen
GASP	General Activity Simulation System	United Steel Corporation (USA)	FORTRAN	diskret	ereignis-orientiert	auf mittleren Anlagen lauffähig
GPSS	General Purpose Simulation System	IBM (USA)	—	diskret	ereignis-orientiert	Programmierung in Blöcken
SIMULA	Simulation Language	Norwegian Computing Centre (Norwegen)	ALGOL	diskret	aktivitäts-orientiert	ALGOL ist eine Untermenge von SIMULA
SIMSCRIPT	Simulation Scripture	Rand Corporation (USA)	FORTRAN	diskret	ereignis-orientiert	SIMSCRIPT II erlaubt selbst-dokumentierende Programmierung
ESI 1	Echtzeit Simulator	WZL (TH AACHEN)	Assembler	quasi-kontinuierlich in Echtzeit	perioden-orientiert	Kleinrechner als Prozesssimulator, mit Fertigungsrechner gekoppelt

GASP II - Systemprogramme

GASP	Steuerung des Simulationslaufes
DATAN	Einlesen der GASP - Parameter
FILEM	Eintragen } von { Einheiten und Ereignissen
REMOVE	Austragen } In/aus Wartebereichen
COLCT	Sammeln
TMST	statistischer Daten
HISTO	Berechnung statistischer Werte
SUMRY	Suchprogramm
FIND	Diverse Zufallsgeneratoren
PROG I	Fehlerrückmeldung
EXIT	Zwischenergebnisse drucken
MONTR	GASP - Datenorganisation
SET	Diverse Druckprogramme für die Auswertung
PROG k	Aktivieren von Ereignissen
EVNTS	

Problemspezifische Anwenderprogramme

MAIN	Hauptprogramm
EVENT n	Diverse Ereignisprogramme
OUTPUT	Spezielles Druckprogramm

GASP II- und Anwenderprogramme

Bild 2:
Unterprogramme des GASP-Simulationspaketes

Bild 3:
Prinzipieller Ablaufplan einer GASP-Simulation

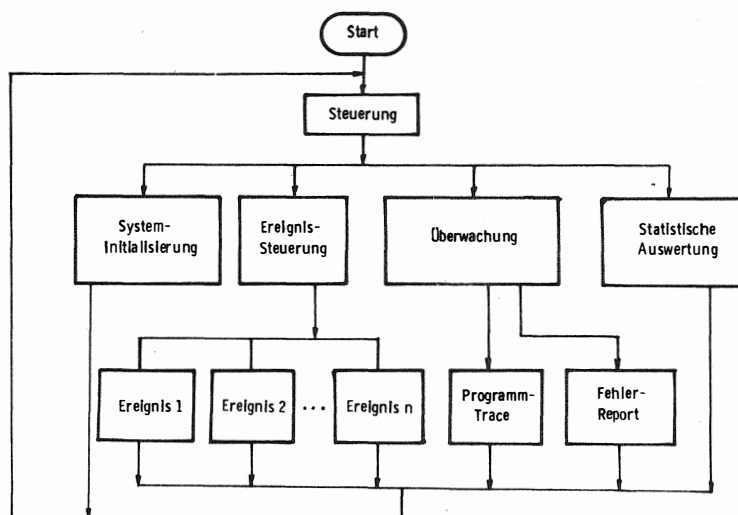


Bild 4:
Matrix-Darstellung der Programmverzahnung

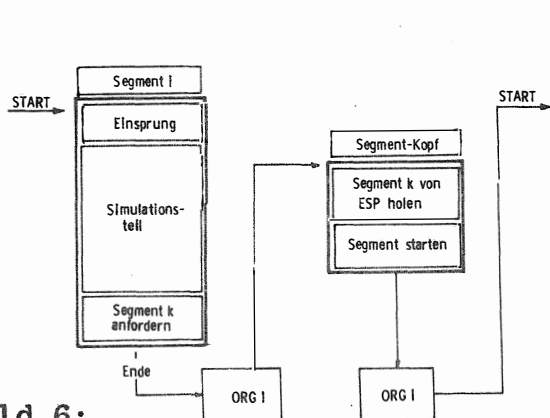
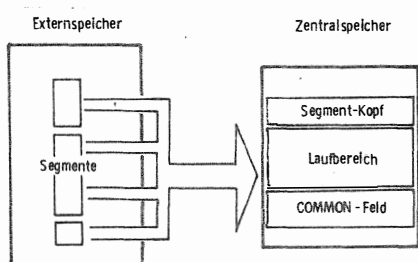
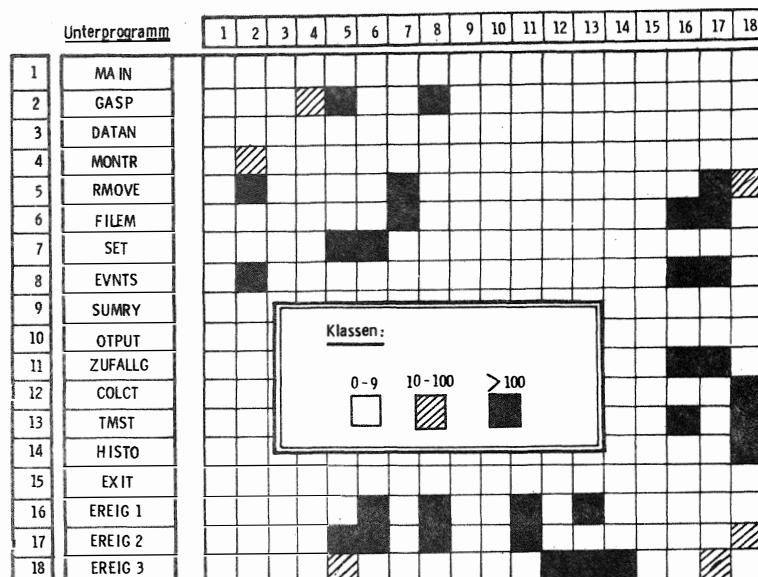


Bild 6:
Organisation der Segmentwechsel

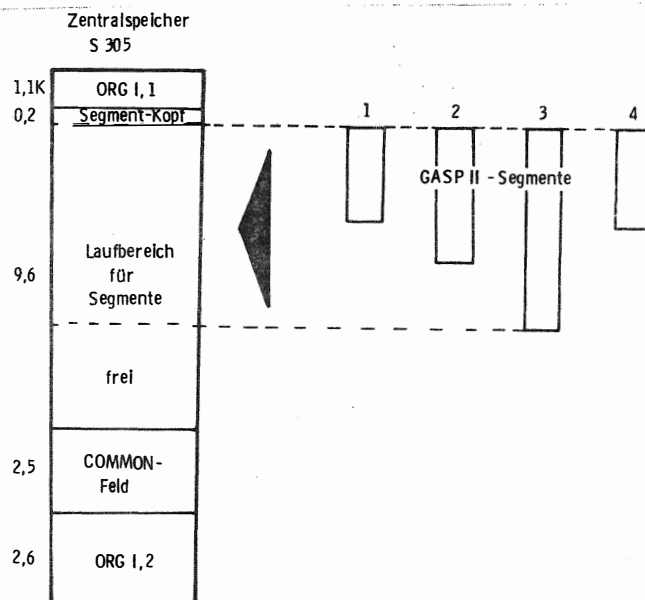


Bild 5:
Segmentierung des GASP-Paketes

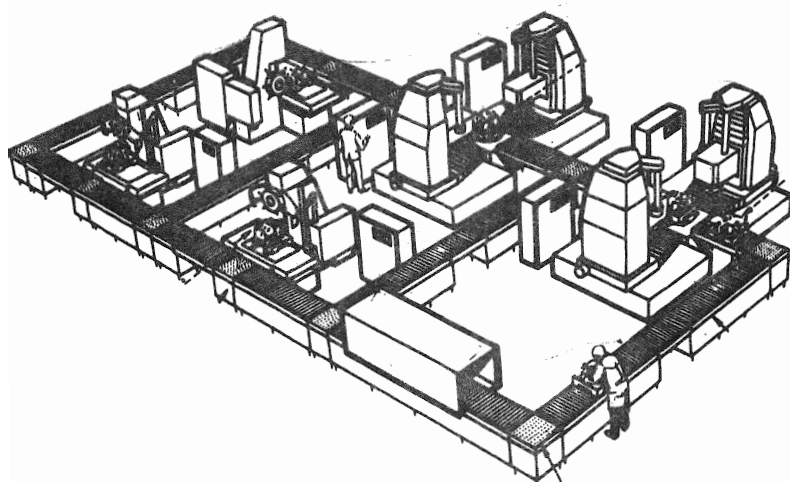


Bild 7: Modell eines flexiblen Fertigungssystems von Cincinnati