HOKKAIDO UNIVERSITY

Title	会話型実時間離散系シミュレーション・システム"RISS"
Author(s)	牧野, 圭二
Citation	北海道大學工學部研究報告, 129, 57-68
Issue Date	1986-01-31
Doc URL	https://hdl.handle.net/2115/41961
Type	departmental bulletin paper
File Information	129_57-68.pdf

Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

北海道大学工学部研究報告
第129号（昭和61年）

Bulletin of the Faculty of Engineering,
Hokkaido University, No. 129 (1986)

会話型実時間離散系シミュレーション・システム“RISS”

牧野圭二

（昭和60年9月30日受理）

“RISS”: An On-line Real-time Interactive Discrete-System Simulator

Keiji MAKINO

(Received September 30, 1985)

Abstract

This paper describes a GPSS-type system simulator RISS, which is a subsystem of the multi-purpose simulator HOSS at the Hokkaido University that is open to researchers. RISS is a real-time interactive system simulator. It is characterized by a trial-and-error process improved by an on-line simulation technique and a man-machine communication in the process of simulation. The hardware architecture is a master-slave system consisting of a master processor and a slave processor connected with a DMA interface, and the latter is furnished with a special simulation console panel for RISS. The slave porforms computation in the simulation controlled by linear simulation time based on its CPU clock. Input/output and control operations are separated from the description of the program body and are organized as a series of system commands issued in execution. A command menu guides the user of RISS.

1. はじめに

RISS (Real-time Interactive System Simulator) は，北海道大学全学共同利用施設汎用シミュレータ施設に設置され稼動中である全ディジタル式汎用シミュレータ「高速システム・シミュレーション装置（HOSS)」の離散系シミュレーション用のサブシステムである。

旧システムの時代には主機がハイブリッド計算機であったため連続系のシステムしか取り扱うことができなかったが，HOSSの計画[2]においてはその主目的たる連続系シミュレータICOSSの開発並びに全ディジタル化に際し，汎用シミュレータ施設として今後の多様なシミュレーションへの対応の準備の必要性から本サブシステムRISSの開発が同時に計画され実現された。[3] 特に将来における連続系と離散系との混合系のシミュレーションや大規模離散系のシミュレーションなどへ対応していくために，ICOSSの並列演算装置PPAでの離散系シミュレーション・プログラ

全学共同利用施設　汎用シミュレータ施設
Simulation Center

ムの並列処理化を含め，それらのシミュレーション技法あるいはシミュレータの研究にも役立つよう考慮が考慮が払われている。[8] 従って，RISS は当初からその処理実行体系において連続系シミュレータ ICOSS との整合性，並行性に重点を置き，シミュレータとは情報処理装置としての性格・機能を有する模擬実験装置であるとの同一の設計思想のもとに，会話型，実時間系のシミュレータとして設計されている。

2. RISS の特徴

離散系シミュレーションでは対象システムにおける時間の経過に伴って生起する状態変化がそれぞれ瞬間的かつ離散的であると仮定し近似を行ってその記述，分析を行うが，そのモデル化の仕方により多くのタイプの言語がある。RISS では一般性と汎用性とから，その記述言語仕様を乱数発生による待ち行列を中心としたイベント型の GPSS(General Purpose System Simulator)[1] に準じたものとしている。ただし通常の汎用大型計算機などのバッチ型 GPSS と比較すると，扱える問題の規模や組み込まれているプログラム・ブロックの種類等は一回り小さいものとなっている。

RISS では，プログラムの作成から実行に至るまでの処理の流れを会話形式で十分制御できるように，処理の流れを，大きく，モデル化フェーズ，編集フェーズ，実行フェーズの三つのフェーズに分けると共に，編集フェーズで作成するプログラム本体には純枠にモデルのみを記述するようにして実行制御から完全に分離，独立させている。従って，実行フェーズではプログラムを再翻訳することなくシステム・コマンドのみで解の表示や実行環境の変更をし再試行することができる。このとき，実行状態を直接観察し制御できるように RISS 専用のシミュレーション・コンソールを用意している。また，シミュレーションの実行が他の利用者や他の処理から独立した状態で行えるように演算プロセッサを独立させ，実時間あるいはリニア・タイム・スケールでのシミュレーションを実現している。

利用者からみた RISS の持つ主な特徴は次のようにまとめることができる。

1) インタラクティブなシミュレーションの実行機能　実行中の時間スケールの変更，実行の任意時点で実行を中断して，モデルの状態の観察，モデルの状況に関する統計量の調査，パラメータ等シミュレーションの条件の変更が自由にできる。

2) リニア・タイム・スケール（実時間）シミュレーションの実行　シミュレーションの実行はハードウェア・クロックをベースにしている。

3) シミュレーション実行の高速化　汎用のホスト・プロセッサに離散系シミュレーションの実行専用のスレーブ・プロセッサを DMA 転送のインタフェイスで結合した階層構造にしシミュレーション計算の実行の独立を図っている。しかも利用者はこのハードウェア構成を意識する必要がない。

4) オンライン・デバッグ機能　シミュレーション・プログラムを1ステップずつ実行できるモードが有りそのつど状態をチェックすることができる。また，実行中状態がシミュレーション・コンソールに表示されておりシミュレーションの流れを容易に把握できる。

5) プログラム作成の容易性　プログラム言語は GPSS 系統でありシミュレーション・モデルを記述しやすい。ソース・プログラムの作成と翻訳処理はシミュレーションの実行と並行して行うことができる。

6) シミュレーション実行の柔軟性　シミュレーション実行の制御手段としてシミュレーショ

ン・コンソール上の専用コマンド・キーと通常のキー・ボード（ソフトウェア・コンソール）とを選択できる。

3. RISS のシステム構成

3. 1 ハードウェア構成

RISS のハードウェア・システムはマスタ-スレーブ方式を採用し，階層構造を持った機能分散型の構成である(図 1)。マスタ・プロセッサは，HOSS の主計算機レベルのホスト計算機 DEC 社製 32 ビット・ミニコンピュータ VAX-11/780 を HOSS の他のサブシステムと共用する。スレーブ・プロセッサ(離散系シミュレーション・プロセッサ)は離散系シミュレーション専用とし HOSS の御御プロセッサ・レベルに位置づける。このプロセッサは HOSS の他の演算用あるいは制御用プロセッサと同じく，DEC 社の PDP-11/34 と同等の性能及び類似の機械語命令（マクロ命令）体系を有する三井造船㈱製のマイクロプログラム演算制御方式を用いた 16 ビット・ミニコンピュータ MAP-16 である。

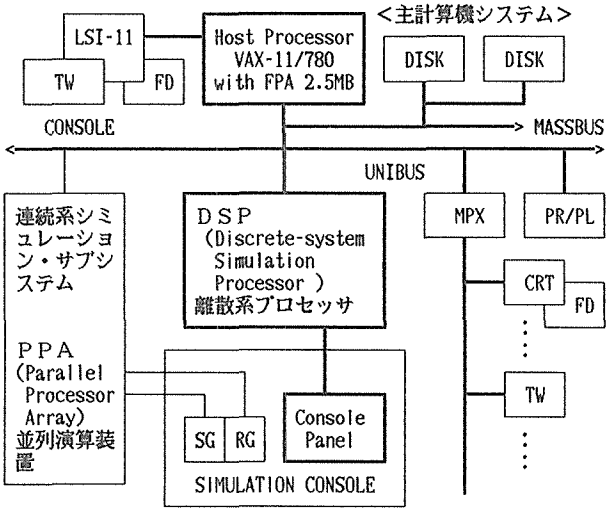

図 1　システム構成

これら両プロセッサ間の処理の分担は次の様である。

マスタ・プロセッサ（マルチ・ジョブ）：実行の前処理，統計量の表示及び実行状態の変更

- ユーザ・プログラムの作成・編集
- ユーザー・プログラムの翻訳
- ユーザー・プログラムの選択
- 実行制御プログラム及びユーザ・プログラムのスレーブ・プロセッサへのローディング
- スレーブ・プロセッサとのコミュニケーション
- システム・パラメータのセット
- 実行状態の変更
- 統計量の表示
- シミュレーションの終了

スレーブ・プロセッサ（シングル・ジョブ）：シミュレーションの実行

- タイマの管理
- ユーザー・プログラムの実行
- 実行制御（START, HALT, STEP 等の処理）
- 統計量の積算（ロギング）
- マスタ・プロセッサとのコミュニケーション

さらに，離散系シミュレーション専用のコンソール・パネルを用意し，ホスト・プロセッサを介さずに，スレーブ・プロセッサでの実行を直接監視・制御する機能をまとめている。これらは，シミュレーションの条件や処理の進行状況を表示する液晶ディスプレイ，実行を直接制御できるコマンド・キー，タイム・スケールをダイナミックに変更できるクロック選択スイッチ(TIME SCALE)やクロック調整ホイール等である（図 2)。

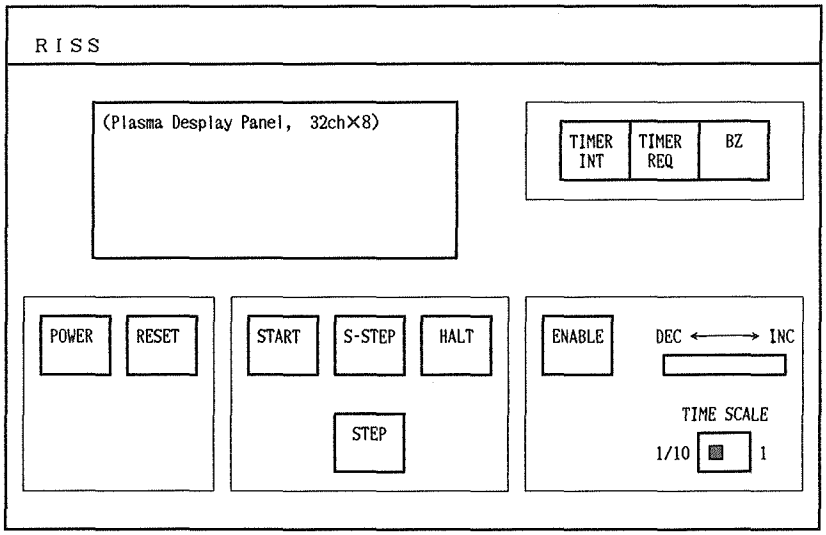

図 2 コンソール・パネル

3. 2 ソフトウェア構成

RISS のソフトウェアは，大きく，エディタ，トランスレータ，実行制御プログラムの三つの部分からなっている。

エディタは RISS 専用エディタであり（付録 I），RISS 完成後利用経験をふまえて利用者の一人である本橋聖一氏と共同で新たに開発したものである。各プログラム・ブロックの記述フォーマットやアーギュメントの使用法などを編集の実行中に参照することができるヘルプ・コマンド，[3] 1 ステートメントを入力する毎にその記述フォーマットのチェックを行い誤りがあれば即座に修正を求めてくるローカル文法チェック機能などが付いている。

RISS トランスレータは，RISS が有する 21 種類のプログラム・ブロック（付録III）を，スレーブ・プロセッサの実行管理プログラム・モジュールが管理・実行可能な形に翻訳，展開しロード・モジュールを作成する（付録II，IV)。

RISS 実行制御プログラムは，シミュレーション・クロックの管理や統計量の収集などを通じて実行を制御すると共に，実行時に利用者が入力するシステム・コマンド（付録V）を解釈し，そ

の種類，すなわち，実行制御コマンド，解表示コマンド，システム・パラメータ・セット・コマンド，終了コマンドのそれぞれに応じ，実行順序の管理，統計量表示，システム・パラメータの設定，終了処理を行う。また実行形式プログラムのスレーブ・プロセッサへのダウン・ロードも行う。実行は任意の時点で中断でき，その時点でシステム・コマンドを入力することができる。いずれのコマンドも，そのコマンド実行後，プログラムを再ロードせず実行を再開できる。

3.3　シミュレーション時間の管理

RISS は実時間型シミュレータであり実時間あるいは実時間に比例したリニア・タイム・スケールでシミュレーションが実行される必要がある。このためシミュレーション時間の管理は，基本クロックをハードウェアで発生させこのクロックを基に一定間隔でシミュレーション・クロックを進める単位時間方式を採用している。以下に示すように，このクロックのスケールはソフトウェアとハードウェアの両者により可変となっている。

1)　タイム・ベース　　タイム・ベースとはシミュレーション実行のためにハードウェアにより発生される基本時間クロックのクロック間隔のことで，CPU クロックをクロック割込発生機構に通すことにより生成している。クロック・レートは 10 μsec と 100 μsec を選択でき，割込みは 1～4095 個の間隔で制御できる。従ってタイム・ベースは 10 μsec から 409.5 msec までの範囲で設定可能であり，その設定はそれぞれ離散系シミュレーション・コンソール上のクロック選択スイッチとクロック調整ホイールとを用いて実行時にダイナミックに行うことができる。

2)　ユニット・タイム　　ユニット・タイムとはシミュレーションの実行で実際に用いられるクロック間隔のことで，タイム・ベースにユニット・タイム係数を掛けたものである。ユニット・タイム係数の値の範囲は 1～32767 であり，実行直前にシステム・コマンドの一つであるシステム・パラメータ・セット・コマンドで設定される。

3)　シミュレーション・タイム　　シミュレーション・タイムはシミュレーション開始時点から経過したユニット・タイムの個数で表す。表現できるユニット・タイムの最大個数は 32767 である。従って，実行時にシミュレーション可能な時間範囲は，ユニット・タイムで正規化した時間で 1～32767 である。

4.　RISS の実行手順

RISS ではモデルの作成から実行まで以下の三つのフェーズに沿って進められる。[3]

(1)　モデル化フェーズ（シミュレーション・モデルの作成）

シミュレータを使用する前の準備段階であり，対象システムの分析や仮定の設定などを通して対象システムをシミュレーション言語で記述できる概念で記述，表現を行う段階である。

RISS のような GPSS 型のシミュレータでのモデル化[1),3)]は，対象システムを大きくその対象システムの有する主に時間的なサービス機能を表す静的な要素と，このシステム内を通過しながらサービスを受けていく物を表す動的な要素とに分けて考える。その上で動的要素が静的要素からサービスを受ける順にその間の関係をブロック・ダイアグラム（機能ブロック図）で表す。

(2)　編集フェーズ（シミュレーション・プログラムの記述と入力）

モデル化フェーズで作成された対象システムのモデル記述を実際にプログラムとして入力し，実行可能な形に翻訳を行う段階である。

ソース・プログラムの入力，編集は RISS の専用エディタで行う。RISS の各プログラム・ブロック文は，一般にプログラム・ブロック番号部，オペレーション・コード部，アーギュメント部，

注釈部の四つの部分からなり，1行で記述される。モデルからプログラムへの変換は，ブロック・ダイアグラムの構成要素とプログラム・ブロックのオペレーション・コードとが一対一に対応付けられており容易である。[3]

プログラムの最終行としてEND文を加えた後，トランスレータを起動して実行形式プログラムへの翻訳，交換を行う。このときプログラムのグローバル文法チェックが同時に行われる。

(3) **実行フェーズ**（プログラムのロードとコマンドによる実行）

編集フェーズで作成された実行形式プログラムを実際にプロセッサへロードし，システム・コマンドを用いて実行環境を設定し実行を行う段階である。

実行形式プログラムをスレーブ・プロセッサへダウンロードした後，システム・パラメータ・セット・コマンドで実行に必要な各システム・パラメータの初期値を設定する。

その後実行制御コマンドで実行を進めるが，実行の制御はコンソール切換コマンドで切り換えることによりユーザー端末から行うこともできる。実行の仕方はプログラムの完成度などに応じて次の3通りの実行方法が選択できる。

a シングル・ステップ・モード：1プログラム・ブロックの実行終了毎にホールト

b ブロック間実行モード：ウェイト・プログラム・ブロックに達する毎にホールト

c 制限条件無しの実行モード

またシミュレーション実行中の任意の時点でシミュレーションを中断（ホールト状態）でき，この状態においてすべてのシステム・コマンドが実行できる。従ってその時点で解表示コマンドを用いるとそれまでの実行結果やその時のモデルの状態などを表示させることができる。実行時におけるシステム・コマンドによるシステムの状態の遷移を図3に示す。

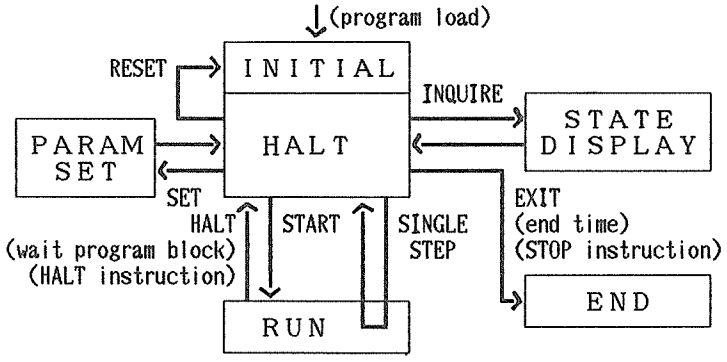

図 3 システム状態の遷移

5. おわりに

本論文では，北海道大学全学共同利用施設汎用シミュレータ施設で設計，開発し，現在設置され稼動中である全ディジタル式汎用シミュレータHOSSのサブシステム，会話型実時間離散系シミュレータRISSの概要について述べた。

従来，計算機の能力の発展にもかかわらず，ディジタル・シミュレータの開発，改良が当初のシミュレータ・モデル開発時のCPU効率を重視したバッチ処理方式のまま，操作性よりもモデル記述の面にのみ向けられる傾向があった。[1] RISSはこの点の反省からオンライン処理化を進め，離

散系シミュレータとしては他に例をみない，利用者の側に立った会話型，実時間系のシミュレータとして実現されている。

RISSでは，ハードウェア構成においてマスタースレーブ方式を採用すると共にシミュレーション・タイムもハードウェア・クロックに基づくリニアなものとし，実行操作はコマンドに主体を置き，メニュー選択，ガイドライン表示方式を採用している。処理実行体系は，HOSSのサブシステムとして同様な立場にある連続系用のICOSSと同一の設計思想を貫き，並行性，整合性を持たせている。このこともあって，RISSはICOSSの並列演算装置PPAを利用した離散系シミュレータの並列処理化や処理方式の研究などにも活用されている。[5)~7)]

謝　　辞

RISSは，1978年度及び1979年度の2年度にわたる文部省特別設備費によって設置された北海道大学「高速システム・シミュレーション装置」HOSSのサブシステムである。設計，開発から稼動まで長期にわたり検討，協力いただいた汎用シミュレータ施設利用開発小委員会；三井造船㈱システム本部を始めとする関係各位に深く感謝致します。

上記小委員会の委員として熱意を傾けられた故小山昭一先生（当時北海道大学工学部精密工学科助教授）に謹んで哀悼の意を表します。

参 考 文 献

1) 中西俊男：システム・シミュレータ，(1969)，p. 276，産業図書
2) 牧野圭二，小山昭一，三木信弘：信学技報，Vol. 80，(1980)，No. 75，p. 55～66
3) 牧野圭二：汎用シミュレータ室だより(北海道大学全学共同利用施設汎用シミュレータ施設)，No. 11，(1981)，p. 17～35
4) 牧野圭二，小山昭一，本橋聖一：昭和56年電気四学会北海道支部連合大会講演論文集182，(1981)，p. 219～220
5) 石動善久，本橋聖一，牧野圭二，小山昭一：第14回計測自動制御学会北海道支部学術講演会論文集38，(1982)，p. 75～76
6) 後藤環，土肥俊，石動善久，牧野圭二：第15回計測自動制御学会北海道支部学術講演会論文集4，(1983)，p. 7～8
7) 後藤環，土肥俊，石動善久，牧野圭二：北海道大学工学部研究報告，第119号，(1984)，p. 63～73
8) 石動善久，牧野圭二，土肥俊：北海道大学工学部研究報告，第120号，(1984)，p. 121～130

付録Ⅰ　テキスト編集コマンド一覧

#ADD	*n*	行番号 *n* の文の後に文を挿入する文挿入モードになる。
#CHG	*n*	行番号 *n* の文を再入力する。
#DEL	*l* 〔, *m*〕	行番号 *l* 〔から *m* まで〕の文を削除する。
#DEND		"#ADD" による文挿入モードを終了する。
#DISP	*l* 〔, *m*〕	行番号 *l* 〔から *m* まで〕の文をCRT端末に表示する。
#END		現在のプログラムについてテキスト編集を終了する。
#HELP	〔*block-name*〕	ブロックの種類，記述フォーマットなどをCRT端末に表示する。
#LIST	〔*l*, *m*〕	プログラムのリストをPR/PLに出力する。
#MOVE	*l*, *m*, *n*	行番号 *l* から *m* までの文を行番号 *n* の文の後へ移動する。

付録Ⅱ　言語仕様

1　文の種類

1. 1　プログラム・ブロック文（シミュレーション・モデルの各ステップを記述する文）

（一般形）　P*xxxx*　　*OP-CODE*　　*ARG1, ARG2, …, ARGn ;*　〈任意の文字列〉

プログラム・ブロック番号部：先頭に"P"を付けた数字列（文番号に対応し必ず必要）

オペレーション・コード（インストラクション）部：機能を表す名前

アーギュメント部：実行に必要な情報（一般に，次に実行するプログラム・ブロックの番号を含む）

注釈部：セミコロンで始まる任意の文字列（省略が可能）

1. 2　END文（プログラムの終りであることをシステムに指示する文）

（形式）　　END

1. 3　注釈文（単なる注釈であり実行にはなんら影響を与えない文）

（一般形）　;〈任意の文字列〉

必ずセミコロンで始めなければならない。

2　プログラムの構造

（一般形）　P*xxxx*　GENE　G, T, R, P*yyyy*, A　　;〈任意の文字列〉

〈

END

プログラムは，一般にプログラム・ブロック文のGENERATE文で始まり，プログラム・ブロック文及び注釈文が続き，必ずEND文で終らなければならない。また，各文ともに継続行は許されず80文字以内でなければならない。

付録Ⅲ　プログラム・ブロック一覧（ニモニック・コードは先頭の3文字でのみ識別する）

（ニモニック・コード）		（意　味）
ADVA	(advance)	トランザクションを指定時間滞在させる。
ASM	(assemble)	SPLITしたトランザクションを集合し一つにまとめる。
CAN	(cancel)	指定したトランザクションを消滅させる。

CIN	(check in)	セクションの入口を定義する。
CNTR	(counter)	カウンタの内容を更新する。
COMP	(compare)	二つの指定パラメータを比較し，分岐先を決定する。
COUT	(check out)	セクションの出口を定義する。
DBW	(data block)	トランザクション発生の時間分布のテーブルを定義する。
FNC	(function)	パラメータの四則演算，転送を行う。
GATE	(gate)	トランザクションの進行を制御する（待ち行列を作る)。
GENE	(generate)	トランザクションを指定時間分布で発生する。
GOUT	(gate out)	ゲートの出口を定義する。
GSW	(gate switch)	ゲートの開閉を行う。
GTST	(gate test)	ゲートの開閉状態を調べる。
HALT	(halt)	シミュレーションを一時停止し，オペレータに制御を渡す。
NOP	(no-operation)	何も行わない。ダミー・ブロック，分岐先変更等に使用する。
PORT	(portion)	指定確率に従って分岐させる。
SPLIT	(split)	カレント・トランザクションのコピーを行い並行する流れを作る。
SRCH	(search)	トランザクションの中から指定した条件を満足するものを探す。
STOP	(stop)	シミュレーションの実行を終了する。
TERM	(terminate)	カレント・トランザクションを消滅させる。

付録IV　システムの容量

（項　目）	（容　量）	（備　　考）
Program Block	1,000 個	
(Program Block Size	5 kW	実行形式オブジェクト・サイズ)
Tranzaction	600 個	システム内に同時に滞在できる最大個数
Section	100 個	セクションの入れ子は認めない
Facility (Gate)	102 個	
(Capacity of Facility	599 個/gate	一つのゲートに滞在できる最大個数)
Counter	512 個	

付録V　システム・コマンドとその表示項目

1　システム・コマンド（$）及びサブコマンド（#）一覧

1. 1　実行制御コマンド

$START	(start)	シミュレーションの実行の指示
$HALT	(halt)	シミュレーションの実行の一時停止
$RESET	(reset)	シミュレーション・プログラムの初期化
$SSTEP	(single step)	シングル・ステップ・モードでの実行
#〈return〉	(STEP)	シミュレーションの１ステップの実行
#EXIT	(exit)	シングル・ステップ・モードの終了
$OPCON	(operator console)	実行制御コマンドを入力する操作卓の選択
#DC	(discrete-system simulation console; command key)	（ディフォールト）

離散系用専用コンソール・パネルのコマンド・キーの指定

#SC (software console; any terminal)

ソフトウェア・コンソール（任意端末のキー・ボード）の指定

1. 2 解表示コマンド

$INQUIRE (inquire) シミュレーション結果の表示または修正

#XACT	*device, mode*	トランザクション（ザクト）に関する統計量の表示
#SECTION	*device, m, n*	セクションに関する統計量の表示
#GATE	*device, m, n*	ゲートに関する統計量の表示
#COUNTER	*device, m, n*	カウンタ情報の表示
#EXAMINE	*table, No., offset*	共通テーブルの表示
#DEPOSIT	*table, No., offset*	共通テーブルの修正
#EXIT		解表示コマンド・モードの終了

1. 3 システム・パラメータ・セット・コマンド

$SET シミュレーションの実行を制御するシステム・パラメータの設定

#SPB	スタート・プログラム・ブロック（GENEブロックに限る）の番号の指定
#WPB	ウェイト・プログラム・ブロック（ホールトするブロック）の番号の指定
#UTM	ユニット・タイム係数（タイム・スケール）の指定
#STM	スタート・タイム（シミュレーション開始時刻）の指定
#ETM	エンド・タイム（シミュレーション終了時刻）の指定
#EXIT	システム・パラメータ・セット・コマンド・モードの終了

1. 4 終了コマンド

$EXIT	現在のシミュレーション・プログラムの実行を終了
$@	離散系シミュレーション（RISS）の終了

2 解表示コマンドにおける解表示項目

2. 1 解表示装置と表示範囲

device 解表示装置

TT	ユーザー・ターミナル（TT*xx*:）
LP	プリンタ・プロッタ（LPA0:）

mode トランザクション・モード

CEC	Current Event Chain
FEC	Future Event Chain
QUE	Queue Mode
ADV	Advance Mode
MCH	Matching Status Mode
ALL	All Transaction

m, n テーブル番号mからテーブル番号nまで

table 共通テーブル名

SYS	System Parameter Table
XAC	Xact (Transaction) Table
SEC	Section Table

GTE Gate Table
CNT Counter Table

No. テーブル番号
offset 項目番号

2. 2 ザクト（トランザクション）・テーブルの表示項目

1) SCAN indicator 0：スキャン可, 1：スキャン不可
2) DELAY indicator 0：次のブロックへ進める, 1：次のブロックへ進めない
3) CHAIN indicator ザクトのチェイン状態
EMPTY 空テーブル
CEC Current Event Chain
FEC Future Event Chain
QUEUE CECでQueue Mode
ADVAN FECでAdvance
4) FXNO. 先行ザクト番号
5) BXNO. 後続ザクト番号
6) PBNO. 現在のプログラム・ブロック番号
7) BDT(Block Deqarture Time) アドバンスしたザクトが再び動き出す時間
8) QTIME キューに入った時間
9) STIME セクションに入った時間
10) SECNO. セクション番号（1～100）
11) XTIME ザクトの発生時間
12) STS1 ザクトのステータス
GNR ザクトがGENEブロックで生成されてFECにチェイン
ADV ザクトがアドバンス中
F. O カレント・ザクトがファシリテイの外にあって進行中
F. Q ザクトがファシリティーにキュー
F. S ザクトがファシリティーに滞在中
F. A ザクトがファシリティの中でアドバンス中
13) STS2 ファシリティ（ゲート）番号
（特にSTS1がF. Q, F. S, F. A, の場合のみ表示）
14) PRI プライオリティ（0～255）
15) GTIM*n*, GNO*n* （1≦*n*≦4） GTIME ザクトがファクシリティに進入した時間
GNO ゲート番号（ファシリティ番号）
16) MSTS マッチング・ステータス
0：マッチング状態
1：アセンブリ・ブロックで相手側のザクトの到着待ち
17) ASXNO. スプリット・ブロックでセットされる相手側のザクト番号

18) WAIT indicator　　0：ノー・ウェイト，　　1：ウェイト状態
19) PARAn　　(n=1, 2. 3) パラメータ

2. 3 セクション・テーブルの表示項目

1) SECTION NO.　　セクション・テーブル番号
2) TOTAL XACT COUNT　　セクションに入ったザクトの総数
3) MAX. STAY TIME　　ザクトのセクション内での最大滞在時間
4) AVE. STAY TIME　　ザクトのセクション内での平均滞在時間
5) STAY XACT COUNT　　現在セクション内に滞在しているザクトの個数
6) MAX. STAY COUNT　　ザクトのセクション内での最大滞在個数

2. 4 ゲート・テーブルの表示項目

1) GATE NO.　　ゲート番号
2) MAX. QUEUE TIME　　最大待ち時間
3) AVE. QUEUE TIME　　平均待ち時間
4) ZERO ENTRY　　待ち時間なしでゲートを通過したザクトの個数
5) MAX. STAY TIME　　最大滞在時間
6) QUEUE COUNT　　現時点での待ちザクトの個数
7) STAY COUNT　　現時点でファシリティに滞在しているザクトの個数
8) MAX. QUEUE COUNT　　最大待ち個数
9) TOTAL XACT COUNT　　ファシリティに入ったザクトの総数
10) CAPACITY　　ファシリティの容量（1～32767）
11) MAX. STAY COUNT　　最大滞在個数
12) AVE. STAY TIME　　平均滞在個数