港湾技研資料

TECHNICAL NOTE OF
THE PORT AND HARBOUR RESEARCH INSTITUTE
MINISTRY OF TRANSPORT, JAPAN

No. 133 Mar. 1972

キューイング・シミュレーションと汎用プログラムについて
(FORTRAN Ⅳ使用)

黒田秀彦

運輸省港湾技術研究所

キューイング・シミュレーションと汎用プログラムについて（FORTRAN IV 使用）

目　　次

キューイング・シミュレーションと汎用プログラムについて（FORTRAN IV使用）

黒田秀彦*

要　旨

本資料は，港湾，空港等の計画・設計において最近しばしば用いられるようになったシミュレーションテクニックをディジタル・コンピューターで行なう場合に必要とされる種々の手続きプログラムをサブルーチン化し，集大成したものである。

これらのサブルーチンは主として交通問題のシミュレーションを行なう場合の基本的な作業，動作をサブルーチンもしくはファンクション化し，あらゆる待ち合せ問題のシミュレーションをこれらの手続きプログラムのコールの系列だけで簡単に行なえるようにした。そして2，3の例題と共にこれらの使用法を示した。

1. まえがき

港湾の計画・設計をより合理的なものとしてゆくためには，港湾のシステマティックな解析と設計がなされなければならない。しかして，最近この港湾のシステム設計に関しても，オペレーションズリサーチの手法や，数学的モデルによる解析がしばしば用いられるようになってきたが，数学的モデルによる取扱いは数学的モデルがしばしばそうであるように，その対象範囲は非常に限定される場合が多い。そこで，電子計算機の発達と共に，港湾の計画，調査において，計算機によるシミュレーションの手法が，しばしば用いられている。しかし，このようなシミュレーションを電子計算機で行なうために，従来，問題が生じる毎に新たなプログラムを作成し，その作成には多大な時間と費用を費し，かつシミュレーション手法と，プログラミングに習熟していなければ，不可能であったが，今回作成したQSSPは，港湾，空港等のように待ち現象の生じる系（システム）のシミュレーションプログラムを，論理的なフローを作成するだけで容易にコーディングが可能であり，FORTRANIVの入門程度の知識で，複雑なシステム・シミュレーションができるようにしたものである。ここには，このQSSPの概要とその使用法について，若干の例題と共に述べた。

2. シミュレーションとシステム分析

シミュレーションの定義にはいろいろの表現がなされているが，「JIS Z8121 オペレーションズ・リサーチ用語」（1967）では，シミュレーションとは「対象とする体系（システム）についての模型（モデル）による実験」であると定義されている。

システム分析を行なう場合，その構造や機能が複雑であればあるほど，単純な数式や既存のデータの解析程度では，とてもそのシステムの実態を解明し，真の問題点を具体的に捕えることが困難である。これを何らかの代用手段で解決しようとして考えられたのが，このシミュレーション手法である。

最近は複雑，大規模になりつゝある港湾の設計・計画においてもこのシステム分析の手法が必要となってきている。特に複雑な湾内船舶動態のシミュレーション等，計算機の大型化と共に所々で使われるようになっている。しかしながら，デジタルシミュレーション（Digital Computer によるシミュレーション）は，そのプログラミングに多大の費用と時間を費やし，結果とモデルの精度に対する考察はしばしば乱雑になりがちである。そこでより早く，より精度の高いプログラミングが必要となる。

ここではデジタルシミュレーションの方法と精度について，若干の注釈を加えたい。

＊　設計基準部　計算室

デジタルシミュレーションのテクニックは一般に

Time−slicing procedure

Event−sequencing method

の２つのテクニックがあり，対象とするシステムと分析に必要とする精度によって，このどちらかのテクニックを採用する。

一般にシステムの状態は，連続又は非連続な変数によって表現される。非連続な状態変数の例としては，生産工場における「工程の数」が挙げられる。精錬所のような生産過程においては「温度」が連続的に変化する状態変数である。連続的な過程ではシミュレーションによってシステムの状態に関する有用な情報を全て獲得することは不可能である。この場合 sampling の間隔を決定し，時間的に，一定間隔毎に必要な情報を得る方法があるのみである。一方非連続過程においては，方法選択（シミュレーションテクニックについて）を迫られる。すなわち，モデルが Event−sequencing であるか Time−slicing であるかを決定しなければならない。

具体的に港湾におけるシミュレーションのように「待ち行列」の問題をとりあげると，これは非連続過程である。システムの状態はシステム中の客人数（隻数）が特性値である。又，生起事象の構成要素は(1)客（船舶）の到着と(2)客（船舶）の出発である。

Event−sequencing simulation では，プログラムは，ある事象から次の事象へ直接進む。ここではシステムの状態は，いつでも記録され，保存される。事象生起の間隔も同様である。この方法によればすべての情報が求められる。

Time−slicing simulation ではプログラムはシステムの状態をある一定間隔ごとに記録する。ここでは明らかに情報の一部は失なわれる。特に問題となるのは，隣り合うサンプリング間隔において多くの事象が生起する場合である。こゝでは多くの客が高密度で到着し，かつ引き続いて系より立ち去る場合，次のサンプリング時点では「システムに状態変化なし」と記録されることになる。きわめて多くの情報が失なわれたわけである。すなわち Time−slicing 法ではシステムの状態変化を単に近似するにとどまる。この様に両者を比較した場合，event−sequencing の方がすぐれていることは明らかであるが，シミュレーションの費用等の点では time−slicing の方が有利である。

システムシュミレーションの問題には，この方法選択の他に，いかに推定値の精度をよくするかを考えねばならない。推定値の精度を決定する基本的な測度としては一般に推定値の変動（Variance）を考え，次の３点について検討がなされなければならない。

1. 推定値の不偏性（unbiasedness）
2. 推定量の効率（efficiency）
3. 変動縮少のテクニック

Event−sequencing 法の場合，一般に推定量の変動を小さくするためには，次の２つのいずれかによる。すなわち「くり返す」か「全観測時間を延長する」かである。しかし，この場合には，実験をくり返えすことが有利であることが証明されている（文献４）

この他にデジタルシュミレーションをする場合，一般に，推定量の変動がいつ十分小さくなるか，すなわち，いつシミュレーションをやめればよいかを決定することが，大きな問題となるが，この点に関しては決定的な結論は得られていない。したがって，デジタルシミュレーションでシステム分析を行なう場合，できるかぎりプログラミングの時間を節約し，これらの精度の問題により多くの時間をかけるべきであろう。

3. ＱＳＳＰとは

ＱＳＳＰとはQUEUING SYSTEM SIMULATION PROGRAMMES の略で，主として待ち合わせシステムのシミュレーションをDIGITAL COMPUTER で行なう際に必要な手続きプログラム（ＪＩＳ 7000−LEVEL FORTRAN IV に従って作成した）を集大成したものである。

待ち合わせシステムとはいわゆる待ち現象の生じるシステムで，このようなシステムは，従来もTRAFFIC THEORY, QUEUING THEORY, WAITING LINE PROBLEMS などといわれ，数学的定式化がなされてきたが，TRAFFIC THEORY としてもORとしての待ち合わせ理論の取扱いにしても，覆々，数学的取扱がそうであるように，複雑なシステムを記述したり，多くの制御可能変数をもつモデルに対しては極めて無力であることを感じる。特に複雑になりつつある港湾の計画，及びシステム分析にとっては，オペレーション，船舶，陸上輸送機関等の動き，荷役施設，倉庫，上屋等の状態は，単純な数学的記述では，解析しえないことが覆々生じる。

このような複雑なシステムの解析に最も有効であると思われるのが，このシミュレーションの手法であろう。しかしながら，モンテカルロ・シミュレーションを行なうには，机上ではまず不可能に近いと思われるし，計算機で行なうにも，プログラム作成に非常に長い期間を有

し，又，対象システムが変わる毎にプログラムを作り変えねばならない。このような手間を省くために作成されたのが，このＱＳＳＰである。

ＱＳＳＰはシステムを単純なフローで表現することにより，フローそのまゝを記述すれば，シミュレーションプログラムが作成されるように作られている。しかしながら，本プログラムはあくまでFORTRAN をベースに作成されているため，GPSSとかＳIMULAなどのように言語のレベルまでは落とされていない。したがって本プログラムを使用する際には，入門程度の FORTRAN に関する知識を必要とする。しかし，逆に言語レベルにまで枠組を決めていない為，本プログラム群だけでは記述しえないシステムの場合でも，USERがプログラムを追加することが容易であるし，その意味で，GPSS, SIMULA 等より汎用的といえるであろう。

4. ＱＳＳＰのシュレート方法

QSSPはGPSS 等の言語と同様 1. で述べた EVENT SEQUENCE 法に属し，一定の時間間隔ごとに現実のシステムをシミュレートするものではなく，次の最早EVENT（最も早い時間に動く対象）に注目して，シミュレート時刻を更新するVariable length 型のシミュレーションを行なう。

次に簡単な窓口業務を例にとって，ＱＳＳＰの基本になっている EVENT SEQUENCE法によるシミュレーションの仕方を説明する。

例－1

窓口が 2つの STATION1 と 1つの STATION2 が連続しており，到着した客は，初め STATION1 でサービスを受け，次に STATION2 でサービスを受ける。このとき STATION1 が使用中であるとき客はその前に待ち行列を作る。STATION2の前でも同様の状態が生じるとする。このときシミュレーションで，待ちの状態，窓口の利用状況をシミュレートする。

シミュレート途中でシステムの状態が図－1のようになったとする。即ち，呼客番号④は窓口２でのサービスを終了し，システムから去ろうとしている。呼客①は窓口２でサービス中，呼客②③は窓口２のサービスを受けるための待ちに入っている。呼客⑤は窓口１でサービス中呼客⑥は窓口１のサービスを受けようとしている。呼客⑦は，システムに到着しようとしている。

この例のようなシステムをシミュレートする場合，次表－１～表－１７のような表を用意しなければならない。表の説明は次のとおりである。

CEC　：現在時刻においてシミュレートされる対象の呼が入っている。
IZCEC：呼の番号
TA　：呼の行動予定時刻
JJ　：呼の行動番号（種別）
　Ⅰ：窓口１でサービスを受ける
　Ⅱ：窓口２でサービスを受ける
　Ⅲ：システムから出る
DI　：何らかの条件で待ちに入っていることを現わす。（LSONのとき）
DEC　：各種の待ちに入っている呼
　（2, 1）：FACILITY *1の待ち
　（1, 1）：STORAGE **1の待ち
FEC　：シミュレート時刻が進んだとき行動を起す呼
IZFEC：上の呼の番号
FAC(1)：FACILITY１の状態を現わす
　LSONのとき使用可能
　LSOFFのとき使用中
STOR(1)：STORAGE１の残余容量

これらの表を用いて，図－１の状態を表現すると表－１～表－４のようになる。

さてシミュレーションを STEP A の状態から開始してみよう。まず CEC 表の１番目から順に動かしうる呼を探す。４番の呼の DI が LSOFFであるので４番を動かす。４番の行動番号 JJ(4)はⅡであるのでシステムから離脱する。従って CEC表はSTEP A－1 のように変更される。次に STEP A－1のCEC 表を見ると次に６番の呼が動かしうる。６番の行動番号 JJ(6)はⅠで窓口１のサービスを受ける。窓口１のサービス時間を２０分とすると６番の行動予定時刻は TA(6)＝AT（現在時刻）＋２０＝１０：５０＋０：２０＝１１：１０
従って６番は１１：１０分までシミュレーションのコントロールが移るまで行動しないので FEC の表に移す。又窓口１は６番が占有するので STOR(1)＝１－１＝０
従ってCEC，及びFAC(1)，STOR(1)はSTEP A－2の表のようになる。

次にCECにはもう動かし得る呼が無いのでFECから新たに最早行動時刻をもつ呼を探し，シミュレート時刻を更新する。

ここでは呼番号１の行動時刻が最早なのでシミュレー

---

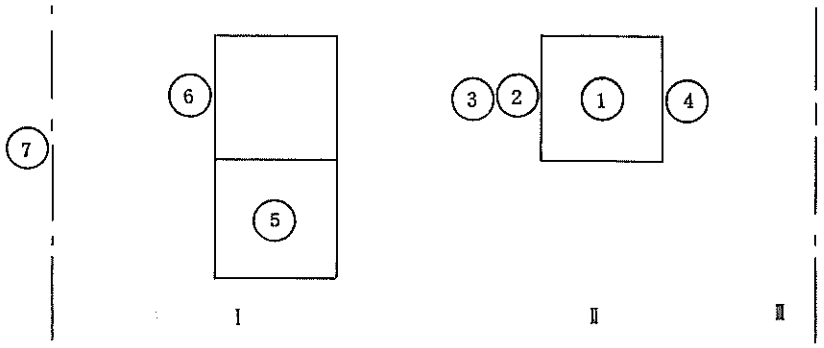
*，**，P22　2. 参照

ト時刻をこの呼１の行動時刻にセットし，（AT=11：00），呼１をCECに移す。呼１は窓口２のサービスを終了し次にシステムから離脱するので，JJ(1)=Ⅱ，また窓口２を開放（FAC(1)=LSON)。従って各表は，表－８～表－１０のようになる。（状態は図－２の状態に移る）

ここで窓口２が開放されたのでⅡにおける待ちから先頭のメンバーである呼２を出し，窓口２のサービスを受けさせる。窓口２のサービス時間を２０分とすると呼２のサービス終了時間は，TA'(2)=AT＋20＝11：00＋20＝11：20　又呼２の窓口２での待ち時間は WT(2)=AT－TA(2)=11：00－10：10＝50分

したがって各表は表－１２～表－１５のようになる。次にFECの中から最早時刻を探し（11：05）その呼をCECに移し，現在時刻を最早時刻にリセット（AT=11：05）（表－１６～表－１７）

以上の操作をシミュレーション終了時刻まで繰り返すと，モデルに組み込まれたシステムの情報は残らず得られることになる。



図－１

STEP A

表－１

CEC	IZCEC	TA	JJ	DI
1	2	10：10	Ⅱ	LSON
2	3	10：10	Ⅱ	LSON
3	4	10：50	Ⅱ	LSOFF
4	6	10：50	Ⅰ	LSOFF

表－２

DEC	2，1	1，1
1	2	
2	3	

表－３

FEC	IZFEC	TA	JJ	DI
1	1	11：00	Ⅱ	LSOFF
2	5	11：05	Ⅱ	LSOFF
3	7	11：07	Ⅰ	LSOFF

表－４

FAC(1)	LSOFF
STOR(1)	1

STEP A-1

表－5

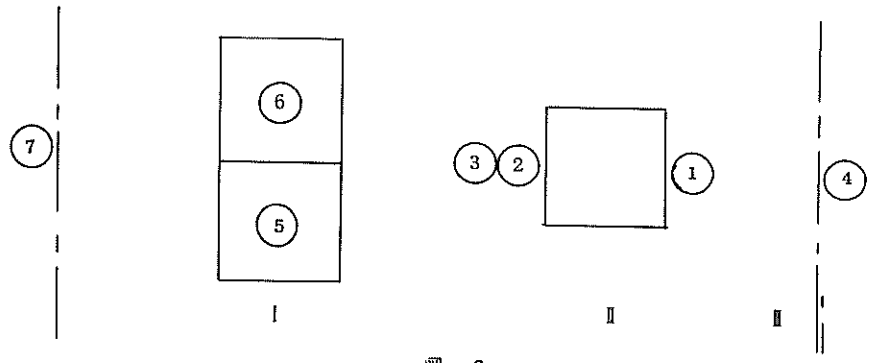
CEC	IZCEC	TA	JJ	DI
1	2	10:10	Ⅱ	LSON
2	3	10:10	Ⅱ	LSON
3	6	10:50	Ⅰ	LSOFF

表－6

CEC	IZCEC	TA	JJ	DI
1	2	10:10	Ⅱ	LSON
2	3	10:10	Ⅱ	LSON

表－7

FAC(1)	LSOFF
STOR(1)	0



図－2

STEP B

表－8

CEC	IZCEC	TA	JJ	DI
1	2	10:10	Ⅱ	LSON
2	3	10:10	Ⅱ	LSON
3	1	11:00	Ⅲ	LSOFF

表－9

DEC	2, 1	1, 1
1	2	
2	3	

表－10

FEC	IZFEC	TA	JJ	DI
1	5	11:05	Ⅱ	LSOFF
2	7	11:07	Ⅰ	LSOFF
3	6	11:10	Ⅲ	LSOFF

表－11

FAC(1)	LSON
STOR(1)	0

表－12

CEC	IZCEC	TA	JJ	DI
1	3	10:10	Ⅱ	LSON

表－13

DEC	2, 1	1, 1
1	3	

表－14

FEC	IZFEC	TA	JJ	DI
1	5	11:05	Ⅱ	LSOFF
2	7	11:07	Ⅰ	LSOFF
3	6	11:10	Ⅱ	LSOFF
4	2	11:20	Ⅲ	LSOFF

表－15

FAC(1)	LSOFF
STOR(1)	0

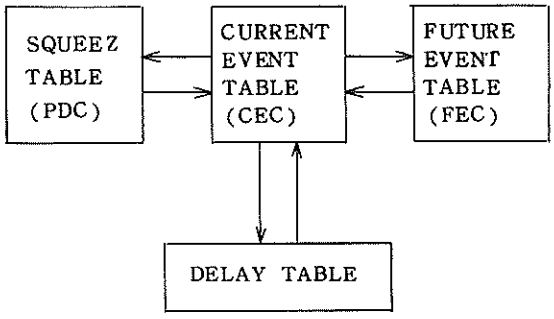
表－16

CEC	IZCEC	TA	JJ	DI
1	3	10:10	Ⅱ	LSON
2	5	11:05	Ⅱ	LSOFF

表－17

FEC	IZFEC	TA	JJ	DI
1	7	11:07	Ⅰ	LSOFF
2	6	11:10	Ⅱ	LSOFF
3	2	11:20	Ⅲ	LSOFF

上記の例におけるシミュレート方法を大まかに示すと図－3のようになる。



図－3　Q・S・S・Pコントロールの概要

QSSPのコントロール部（メイン）は結局FUTURE EVENT CHAINにある呼のうち最早行動開始時刻をもつ呼を全てCECに入れ，シミュレート時刻をこの最早行動開始時刻に更新し，次にCECにある呼を順次動かしていく。この際，DELAY INDICATOR DIがONの状態になっているものについてはDIがOFFの状態になるまで動かさない。FUTURE EVENT CHAIN には，最初に呼が発生されたときとADVANCをCALLしたときに登録され，CECから除かれる。そして，シミュレート時刻がこの呼のEVENT TIMEに更新されるまでControlを受けない。

5. QSSPの基本要素

ここではQSSPを構成している基本的な要素（手続きプログラムの種類と呼），およびそれらのシミュレート方法との関連について説明する。

(1) 手続きプログラム

QSSPでは主として待ち合せシミュレーションで生じると考えられる基本的な動作（到着，離脱，サービスを受ける，待つ等）を表わすために33種類のサブルーチン，およびファンクションが用意されている。USERはこれらのサブルーチン名を線で結び，システムをフローチャート化して記述することにより，シミュレーションモデルを作成する。（例2） QSSPのプログラミングは現実のシステムの論理的な流れにそって，このフローチャートを作成することから始まる。フローチャートにおいて，あるサブルーチン名の所から数本の線で分岐するとき，次のステップにいくつかの行動があることを意味しQSSPはその時のシミュレーションモデルの状態に応じ，論理的（YES or NO）にあるいは確率的に（乱数を用いて）に，いずれか1つの行動を選択する。

(2) 呼

QSSPではシミュレーションの中を動くもの（行動時刻を有するもの）を"呼"とよぶ。呼は港湾のシミュ

レーションにおいては，1. 船舶，フォークリフト，艀等は呼であり，又，2. 貨物の動きをとらえるときは貨物が呼であり，3. 交通のシミュレーションでは，自動車，人，信号機等も又呼であり，対象とするモデルによって人物であったり，物であったり，情報であったりする。

フローチャートはつねにこの呼の行動を追う形で組立てられなければならない。すなわちＱＳＳＰは SUBROUTINE GENERAで呼をモデルの中に作り出し，続く各サブルーチンの機能に従って呼を動かすと共に，ＱＳＳＰ内部に貯えられた情報（窓口や待ちの状態，他の信号，規制等の状態）を更新し，SUBROUTINE TERMIN で，動き終えた呼をモデルの中から取り除くことを繰り返して現実のシステムをシミュレートする。

(3) チェイン

ＱＳＳＰでは，呼が入れられるテーブルを"チェイン(Chain)"と呼ぶ。MAINのQSSP-CONTROL 部は，シミュレーションモデル内の全ての呼をその状態によって，次のいずれかのチェインに入れる。（数個のチェインに重って入っていることもある）

1. CURRENT EVENT CHAIN(CEC)
2. FUTURE EVENT CHAIN(FEC)
3. DELAYED EVENT CHAIN(DEC)
4. SQUEEZED EVENT CHAIN(SEC)

ⅰ) CEC

現在シミュレート時刻と同じ時刻(EVENT TIME)をもつ呼又は，それ以前の EVENT TIME をもつ呼が到着の早い順に入れられる。QSSP CONTROL 部は，この Chain に入っている呼だけをテーブルの1番最初の呼から順に1つずつ動かしていく。

ⅱ) FEC

将来動きうる呼が入れられている。サブルーチン ADVANC に入った呼は，サービス終了時刻になるまでこの CHAINに入れられる。

ⅲ) DEC

何らかの条件を満たさなくて，待ちの状態に入った呼が，待ちに入った順序に入れられる。これは条件毎に作られたテーブルである。シミュレート中に，この条件が満足する状態になったときに，この CHAIN から出される。

ⅳ) SEC

これは，サブルーチン ADVANCに入ってサービスを受けている呼が，他の呼により，割り込みをかけられたときに，割り込みした呼のサービスが終るまで入れられる CHAINである。

〔ＱＳＳＰに与える条件と得られる結果〕

一般に次のものを必らずＱＳＳＰに条件として与えなければならない。

ⅰ) 呼の到着時間間隔

1個の呼が到着してから（シミュレーションモデルの中に作り出されてから）次の呼が到着するまでの到着時間間隔の平均値，分布関数の番号，一般分布の場合は分布関係の値と平均値を与える。

ⅱ) FACILITY, STORAGE の番号

シミュレーションモデルの中で扱うすべての窓口（バース，上屋，倉庫，信号等）について，その番号を与える。ＱＳＳＰでは，複数個の呼を同時に処理できる窓口をSTORAGE と呼び，同時に1個の呼しか処理できない窓口をFACILITY, 処理時間0の窓口（信号，夜間，昼間の区別等）をLOGIC SWITCH と呼ぶが，これらについては全て番号を付す。STORAGE についてはその容量（同時に処理できる呼の最大数）をSTORAGサブルーチンで定義する。又優先順位のついている1群の FACILITY をGROUPとして定義することもできる。

ⅲ) 待ち行列番号

ＱＳＳＰではシミュレーションモデルの中にできる待ち行列を QUEUEと呼び，待ち行列に関する統計量（待ち行列長，平均待ち時間等）を必要とする場合に，その行列を表わす番号を行列番号として与えなければならない。

ⅳ) FACILITY, STORAGE での処理時間，及び信号等の LOGIC SWITCHの変更時間

2で与えたものについて，その中での処理時間（サービス時間，信号から信号までの時間）を ADVANCサブルーチンで，一定値，平均値と分布型を与える。

ⅴ) シミュレーションの終了時刻

シミュレーションのランの長さは，時間をコントロールする GENERAサブルーチンと TERMINサブルーチンで決定される。

6. Q·S·S·Pの手続きプログラムとその使用法

Q·S·S·P は３３種類の手続きプログラム（SUBROUTINE 及び FUNCTION）から成立っており，これらの手続きプログラムに適当な引数（ARGUMENT）を与えて CALL することにより現実のシステムがシミュレートされるようになっている。以下これらの手続きプログラムについて，引数の意味及び使い方を簡単なフ

ローチャートと共に説明する。

1) SUBROUTINE GENERA

〔引数〕

NB ： 何番目のGENERAかを示す番号

IGN： 1＝LOGIC, RESET 等と共に用いるときで，このときにかぎりTERMIN と共に用いない。

0＝上記以外に呼客の発生等に使用する場合

ST ： 発生時間間隔の平均値

シミュレーション期間中，変更のある場合はRESETで値を与える。

AT1： 最初の呼の発生時刻

FN ： 発生時間間隔分布

0：指数分布

1：一定分布

2：一般分布

一般分布の時はFUNCTで分布型を指定しておかねばならない。

＊ ： $1100 （メインの1100番）

NCS： このサブルーチンの出てくる順序

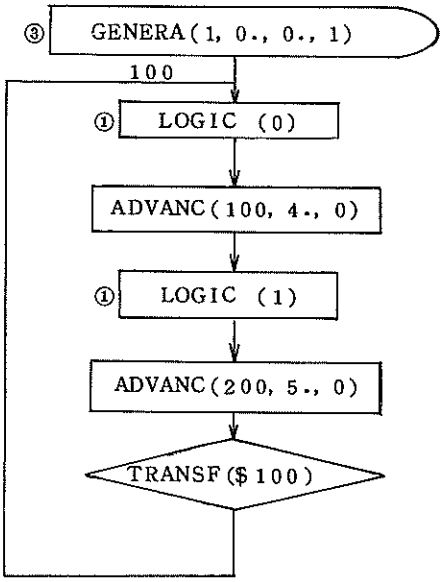
〔仕様〕

各分布と平均値を与えて呼を発生（到着時刻を決定）する。

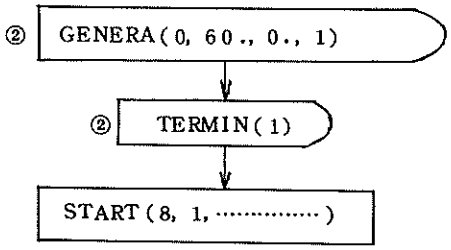
最初の呼の発生は AT1 にセットされ，後メインの1011へ戻る。2回目以降は次に続くサブルーチンを実行する。ROGIC, RESET 等と共に用いて時間のコントロールに使うとき（図－4参照）2回目以降は時間を計算せずに次のサブルーチンを実行する。

シミュレーションの終了をコントロールするときにも使う。（図－5参照）

LOGIC SWITCH （信号，夜昼の区別等）をLSON（信号の場合は青，夜昼の場合は昼の状態）にセットし，この状態を100 クロックだけ持続する。後LSOFF（信号の場合は赤，昼夜の場合は夜）にセットし，この状態を200 クロック持続するということをシミュレーションが終るまで繰り返す。



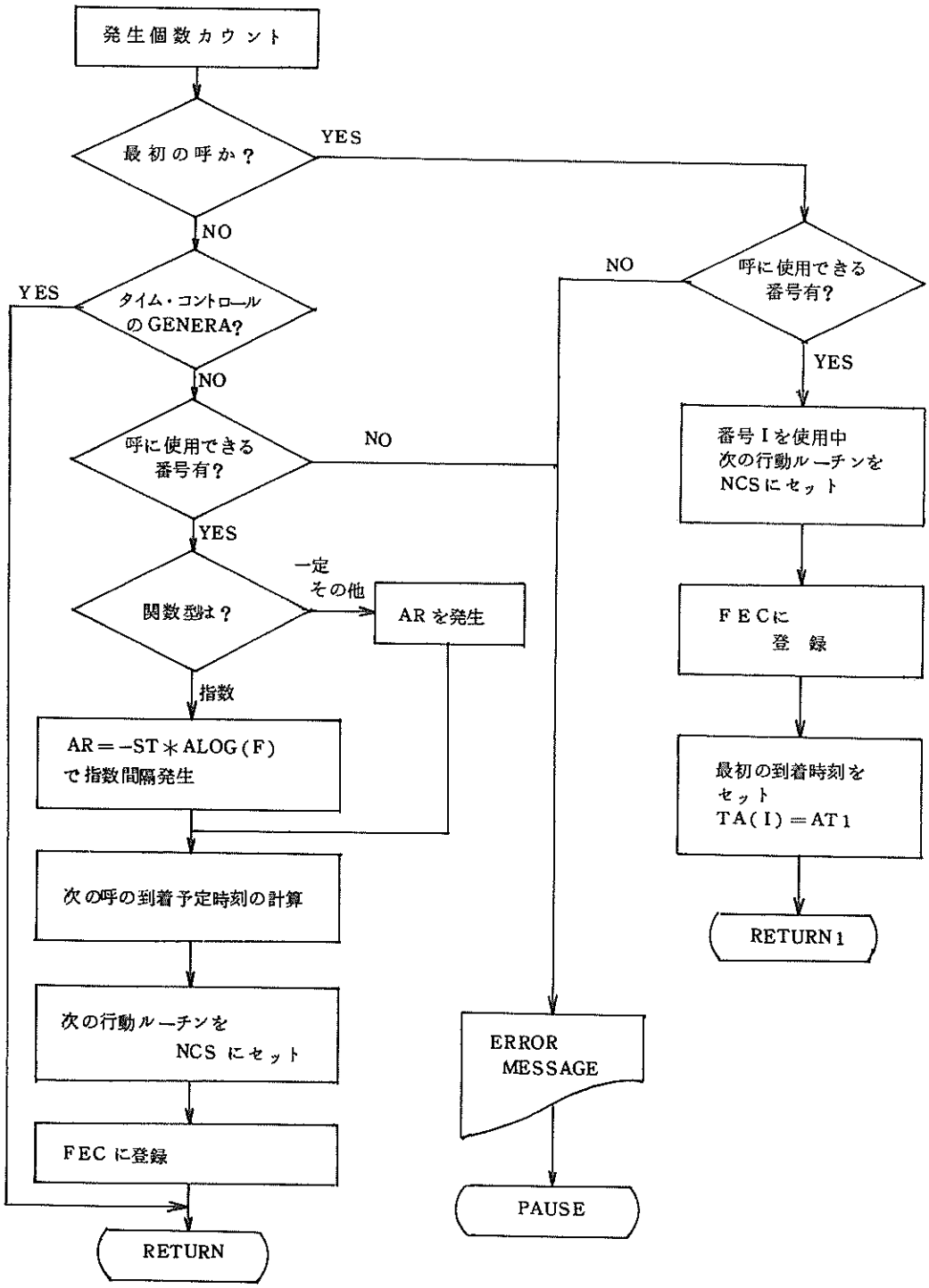
図－4



図－5

60 クロック毎に各種統計量をアウトプットし，480クロックまでこれを繰返し，シミュレーションを終了する。

〔フロー〕


発生個数カウント
最初の呼か？
YES
NO
タイム・コントロールのGENERA?
YES
NO
呼に使用できる番号有？
NO
YES
関数型は？
一定
その他
指数
AR を発生
AR＝−ST＊ALOG(F)で指数間隔発生
次の呼の到着予定時刻の計算
次の行動ルーチンをNCS にセット
FEC に登録
RETURN
呼に使用できる番号有？
NO
YES
番号Ｉを使用中
次の行動ルーチンを
NCS にセット
ＦＥＣに
登　録
最初の到着時刻を
セット
TA(I)＝AT1
RETURN1
ERROR
MESSAGE
PAUSE

ii) SUBROUTINE START

〔引数〕

TSTART: アウトプット回数(終了までの回数)

ITIME : アウトプット間隔

MOD(TSTART/ITIME) = 0のときアウトプットする。TSTARTは1回呼ばれる毎にTSTART=TSTART−ITIMEで変更され，TSTART = 0となったときシミュレーションは終了する。

ISS : STORAGEの最初の番号

IES : STORAGEの最後の番号

ISF : FACILITYの最初の番号

IEF : FACILITYの最後の番号

ISQ : QUEUEの最初の番号

IEQ : QUEUEの最後の番号

＊ : $ 1011

＊ : $ 9999

〔仕様〕

GENERA, TERMINルーチンと共に用い，各種統計量のプリントアウトとシミュレーションの終了時刻を指示する。(図−5参照)

STARTルーチンでアウトプットされる情報

(1) 各GENERA, TERMINルーチンでGENERATE又はTERMINATEされた呼の数

(2) STORAGEに関する統計量

a. STORAGEの番号　b. STORAGEの容量

c. 平均占有量　d. 平均利用率　e. 利用呼数

f. 平均占有時間

(3) FACILITYに関する統計量

a. FACILITYの番号　b. 平均利用率

c. 利用呼数　d. 平均占有時間

(4) QUEUEに対する統計量

a. QUEUEの番号　b. QUEUEを出た呼の数

c. QUEUEの最大長さ　d. 統待ち時間

(5) QUEUEの待ち時間に対するヒストグラム

iii) SUBROUTINE QUEUE

〔引数〕

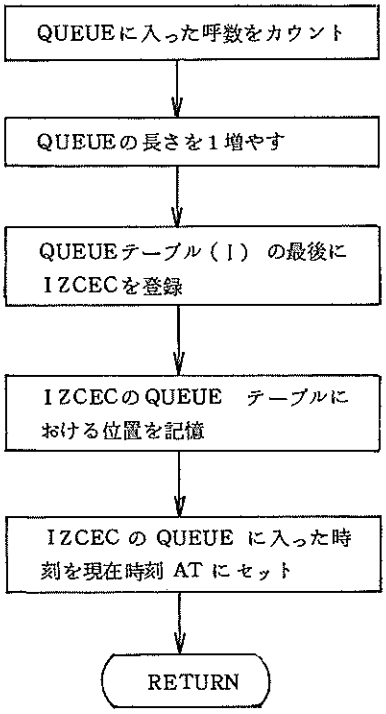
I : QUEUEの番号

〔仕様〕

QUEUEに関する統計量を必要とする場合，必らず指定しないといけない。

次のDEPARTルーチンと対で使用する。(図−6参照)　QUEUEの数は最大10個までである。

〔フロー〕



iv) SUBROUTINE DEPART

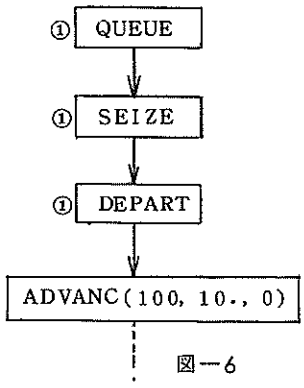
〔引数〕

I : QUEUEの番号

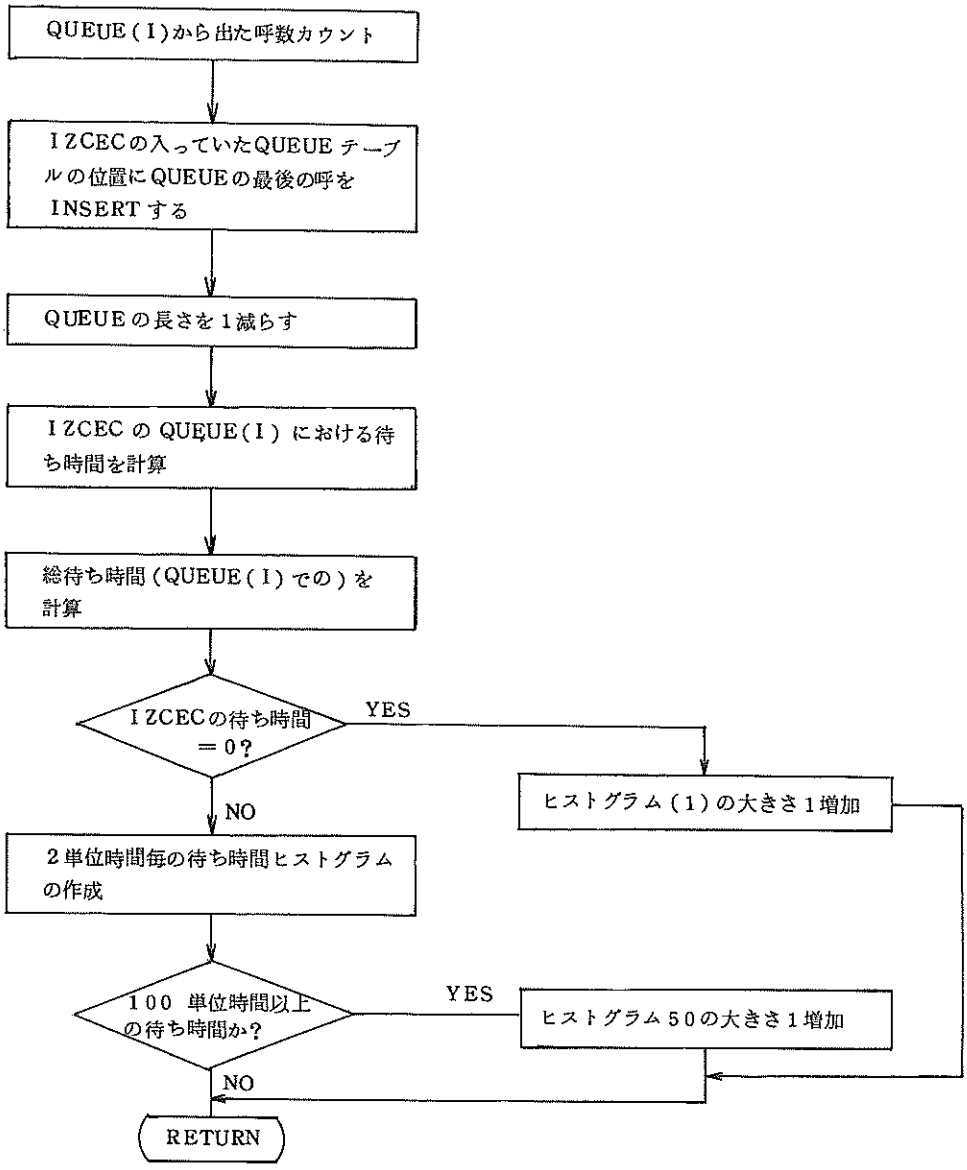
〔使用〕

QUEUEルーチンと対で使用し，QUEUEから出た呼について，そのQUEUEにおける待ち時間，総待ち時間，待ち時間のヒストグラム等を計量する。

例



図−6



V) SUBROUTINE GATE

〔引数〕

SNF : STORAGE か FACILITY か LOGIC かを指定する

＝1～STORAGE

＝2～FACILITY

＝3～LOGIC SWITCH

I : STORAGE, FACILITY 又は LOGICS WITCH の番号

J : STORAGE の場合占有容量

＊ : $1011

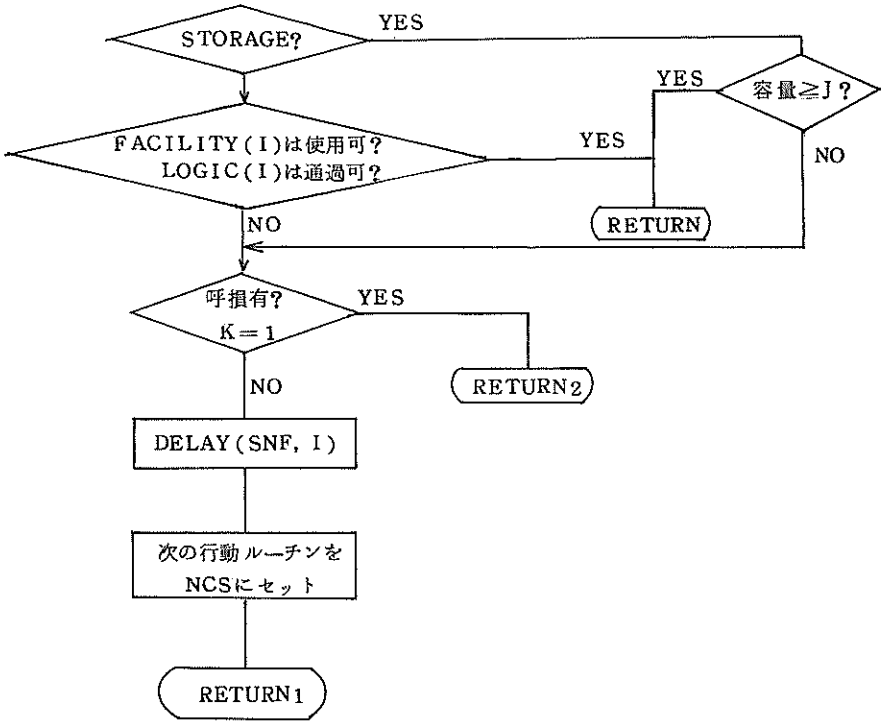
＊ : このグループの TERMIN ルーチンの STATEMENT NUMBER

K : 呼損(待ちに入らずに系から逃げ去る)が

有るか無いかを示す。
＝1～呼損有り
＝0～呼損無し
NCS ：このルーチンの番号
メインの GO TQ (……) nのnと対応する。

〔仕様〕
LOGICSWITCH, FACILITY 又はSTORAGEが通過可能又は使用可能な状態かどうかをチエックする。K＝0の場合，通過不能又は使用不能であれば，LOGIC SWITCH, FACILITY, STORAGEの番号IのDELAY CHAIN に入れる。後，コントロールはメインの１０１１に戻る。K＝1の場合は，呼はDELAY CHAIN に入れないで，２番目の＊で指定した番地へ飛ぶ。普通は呼損を考えて，TERMINのルーチンへ飛ばすことが多い。

〔フロー〕



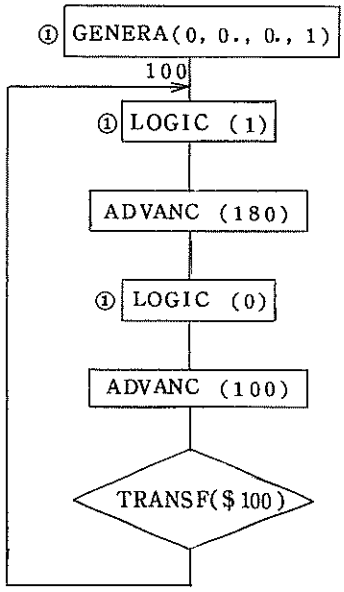
vi) SUBROUTINE LOGIC

〔引数〕
NL ： LOGIC SWITCHの番号
ISR：＝1～SWITCHをLSOFFの状態にセットする。
＝0～SWITCHをLSONの状態にセットする。

〔使用〕
LOGIC SWITCH は交叉点における信号とか夜間規制のある場合等昼夜の区別をしたりするのに使用する。LOGIC SWITCHがLSOFFの状態であれば，呼はそれより前には進めないで，DELAY CHAIN に入るか，呼損する。

交叉点における信号が180クロックだけ赤(LSOFF)の状態であり180クロック過ぎると青(LSON)の状態になり青の状態が１００クロック続く。この状態をシミュレーションが終了するまで繰り返す。

例



図－7

vii) SUBROUTINE SEIZE

〔引数〕

I　：FACILITY の番号

NCS　：このルーチンの番号

TRAN：TRANSA でコントロールしたときに最後に探すFACILITYであるときは1，それ以外は0を指定する。

1番目＊：$1011

2 〃 ＊：TRANSAの STATEMENT 番号

3 〃 ＊：次のSEIZEの番号

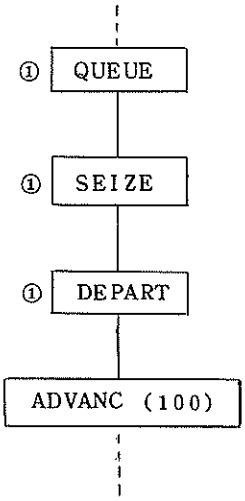
〔仕様〕

FACILITYが空いているかどうか判定し，空いていればそのFACILITYを占有する。空いていなければ，そのFACILITYの為の待ちに入る。

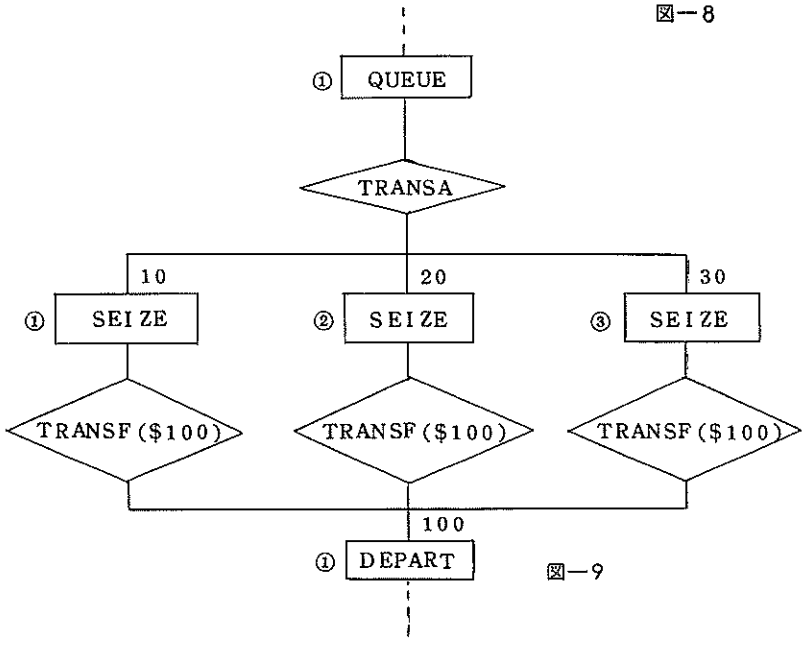
TRANSA でコントロールしているときは最後のSEIZEには第3番目＊に$99999を入れる。

TRANSA でコントロールされていないときは第2，第3＊に$99999を入れる。

例



図－8



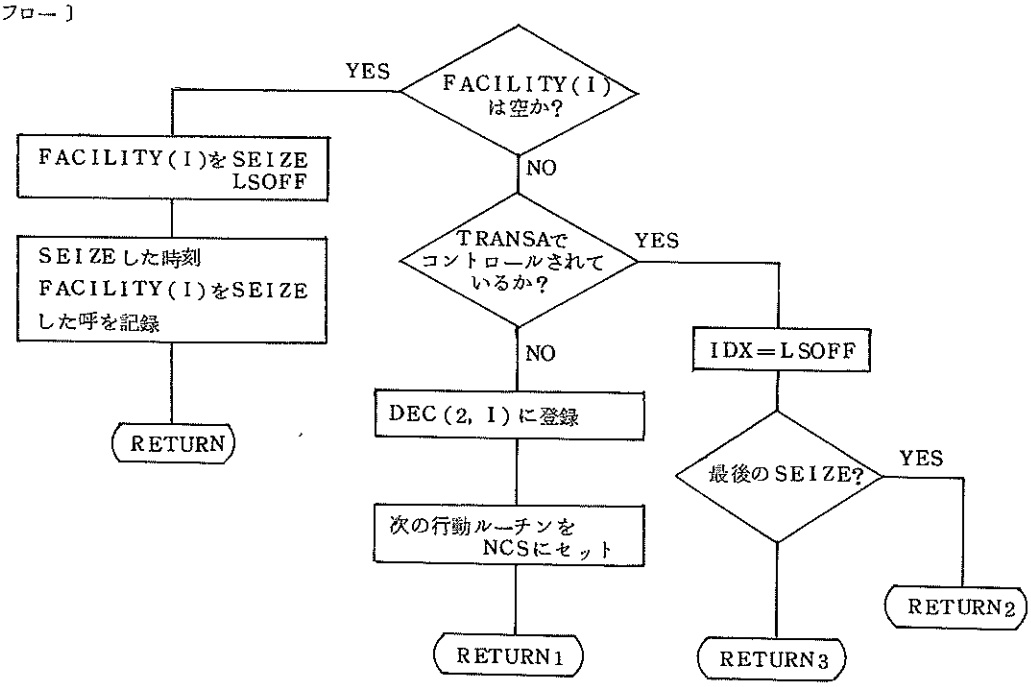
図－9

図－8はFACILITY番号1のFACILITY が空いていればこれをSEIZEし，QUEUE番号1のQUEUEから出て，ADVANC で指定された時間だけここにとどまる。もしFACILITYが空いていなければQUEUE番号1の待ちに入る。

図－9はTRANSA でコントロールされているSEIZEの場合でFACILITY番号1～3まで順に空いているものを探し，もし空いているFACILITYがあればそれをSEIZEしてQUEUE番号1から出る。

もしいずれのFACILITYも空いていなければ，QUEUE 番号1のQUEUEにとどまる。

〔 フロー 〕



viii) SUBROUTINE RELEAS

〔 引数 〕

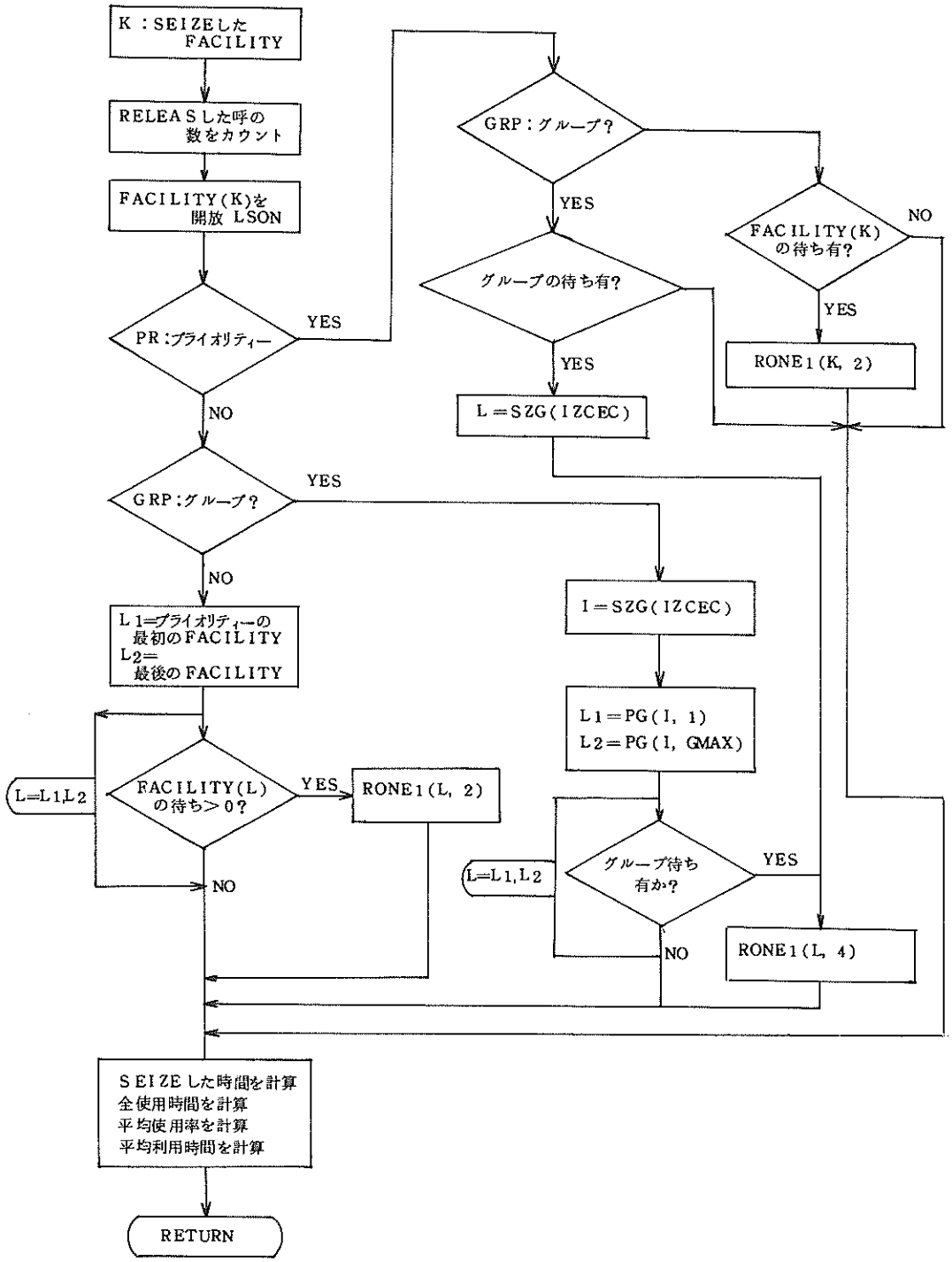
PR ： FACILITYの使用に関してプライオリティーがあるかどうかを指示する。
TRANSA でコントロールされているときは必らずプライオリティーをつける。
＝1～プライオリティー有
＝0～プライオリティー無

GRP ： FACILITYがGSEIZEによってSEIZEされたものかどうかを指示する。
＝1～GSEIZEによってSEIZEされた場合
＝0～SEIZEによってSEIZEされた場合

〔 仕様 〕

SEIZE, GSEIZE と対で用い，SEIZEしたFACILITYを開放し，待ちがあれば，その先頭の呼にFACILITYを開放する。

〔フロー〕


K：SEIZEした FACILITY
RELEASした呼の数をカウント
FACILITY(K)を開放 LSON
PR：プライオリティー
YES
NO
GRP：グループ？
YES
NO
L1＝プライオリティーの最初のFACILITY
L2＝最後のFACILITY
L=L1,L2
FACILITY(L)の待ち＞0？
YES
NO
RONE1(L, 2)
GRP：グループ？
YES
グループの待ち有？
YES
L＝SZG(IZCEC)
FACILITY(K)の待ち有？
NO
YES
RONE1(K, 2)
I＝SZG(IZCEC)
L1＝PG(I, 1)
L2＝PG(I, GMAX)
L=L1,L2
グループ待ち有か？
YES
NO
RONE1(L, 4)
SEIZEした時間を計算
全使用時間を計算
平均使用率を計算
平均利用時間を計算
RETURN

IX) SUBROUTINE GSEIZE

〔引数〕

I　：グループ番号

LAST：TRANSA でコントロールされているとき
＝1
TRANSA でコントロールされていないとき
＝0

＊　：次のGSEIZEの STATEMENT番号
TRANSA でコントロールされていないときは
$ 99999

＊　：TRANSAの STATEMENT番号
TRANSA のコントロールがないときは
$ 99999

〔仕様〕

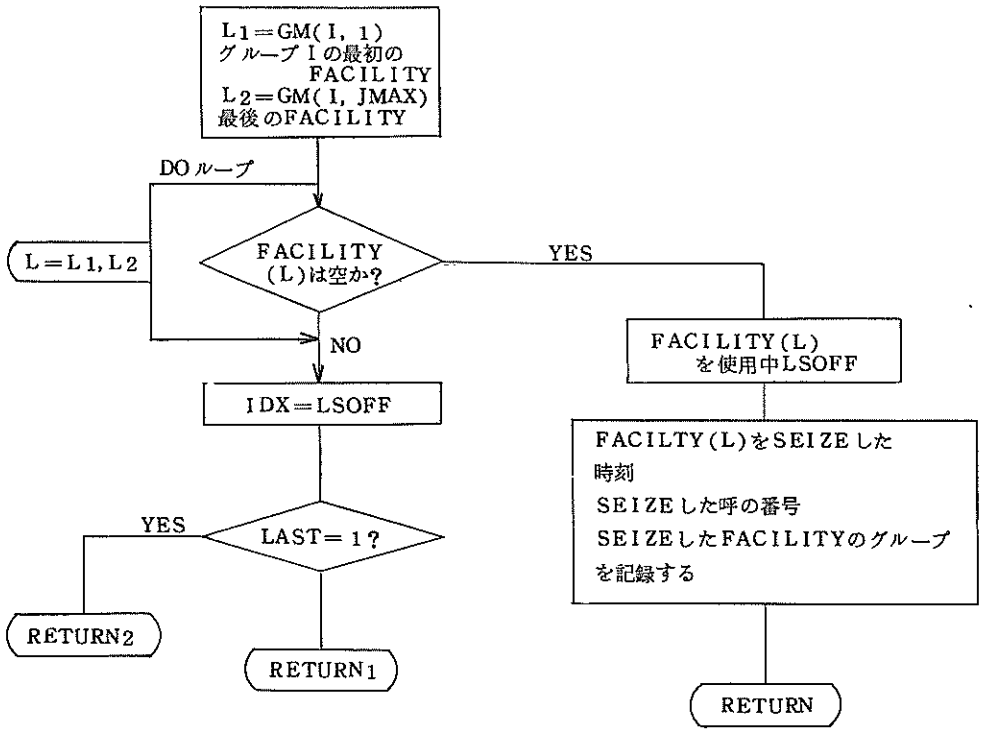
GROUPMで指定したFACILITYを1グループとして，このグループの中で空いているものがあればそのFACILITYを占有する。空いていなければそのグループの待ちに入る。

TRANSAでコントロールしているときは最後のGSEIZEの2番目＊には$ 99999を入れる。

必らずRELEASと対で用いる。

RELEASにはグループの番号を引数に指定する。

〔フロー〕



X) SUBROUTINE ADVANC

〔引数〕

STAD：ADVANCE TIME(サービス時間)の平均値

MCN　：次の行動番号
メインのGO TO(……)，nのnと対応

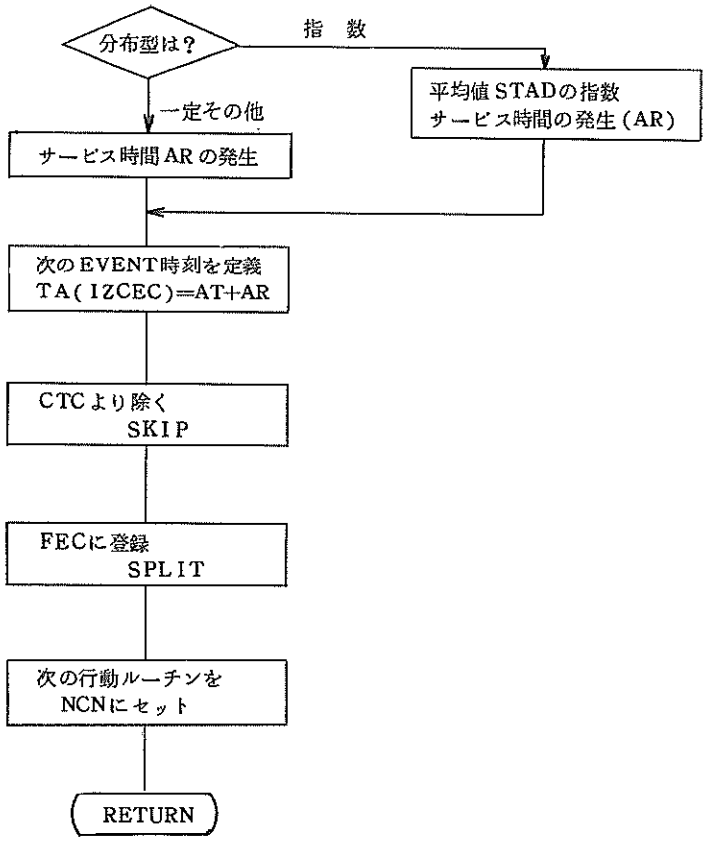
NFC　：ADVANCE TIME(サービス時間)の分布の型
＝0～指数分布
＝1～一定分布
＝2～一般分布

＊　：$ 1011

〔仕様〕

指定した分布型より平均値STADのADVANCE TIME(サービス時間)を乱数により発生し，その間だけこのルーチンにとどまる。

〔フロー〕



(i) SUBROUTINE SIMULA

〔引数〕

KK ：GENERAルーチンの数

〔仕様〕

SIMULATION を初める前にあらゆる変数を初期の状態（FACILITY, STORAGEは空の状態，LOGIC SWITCHはLSONの状態，乱数初期値等）をセットする。プログラムの1番最初にこのルーチンをCALL しなければならない。

(ii) SUBROUTINE SETPAR

〔引数〕

I ：パラメータ番号

Pi ：i＝1～5

パラメータの値を割り当てるための百分率

Ji ：i＝1～5

百分率の値がi番目のパラメータの値

N ：パラメータの値の数

〔仕様〕

与えられた百分率に従がい，呼にパラメータを与える。

例

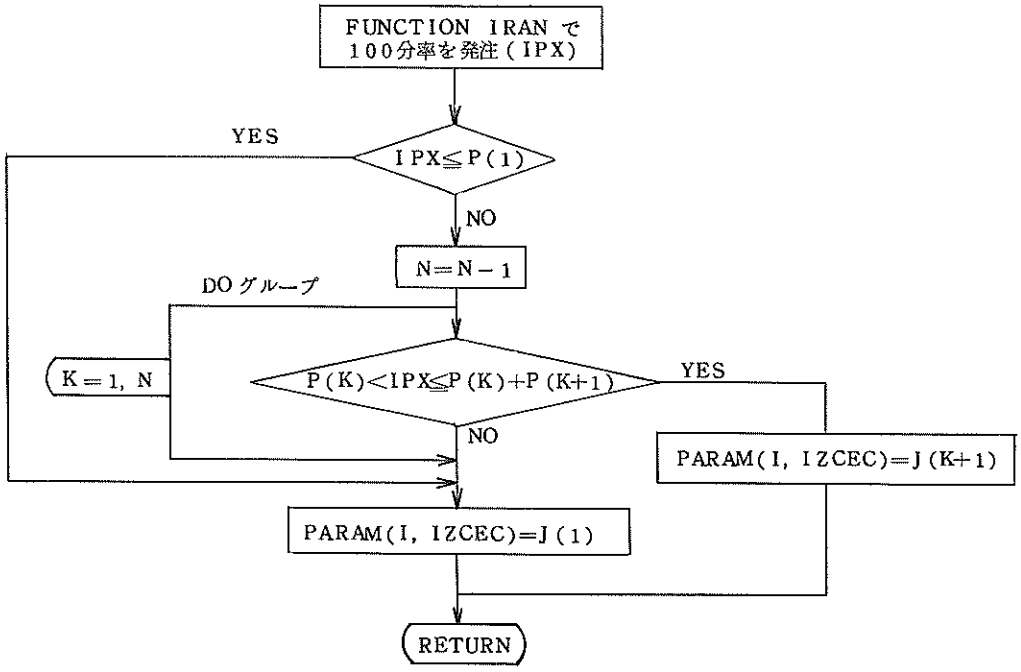
パラメータ番号2に次の百分率に従って値を入れる。

J1＝1　　10％

J2＝2　　15％

J3＝3　　75％

CALL SETPAR(2, 10, 15, 75, 0, 0, 1, 2, 3, 0, 0, 3) 主に次のTRANSP と共に用い，フローを変えるのに用いる。



Xiii) SUBROUTINE TRANSP

〔引数〕

I：パラメータの番号

＊：呼IZCECのパラメータ番号Iのパラメータの値が1のときの飛先番地

＊：呼IZCECのパラメータ番号Iのパラメータの値が2のときの飛先番地

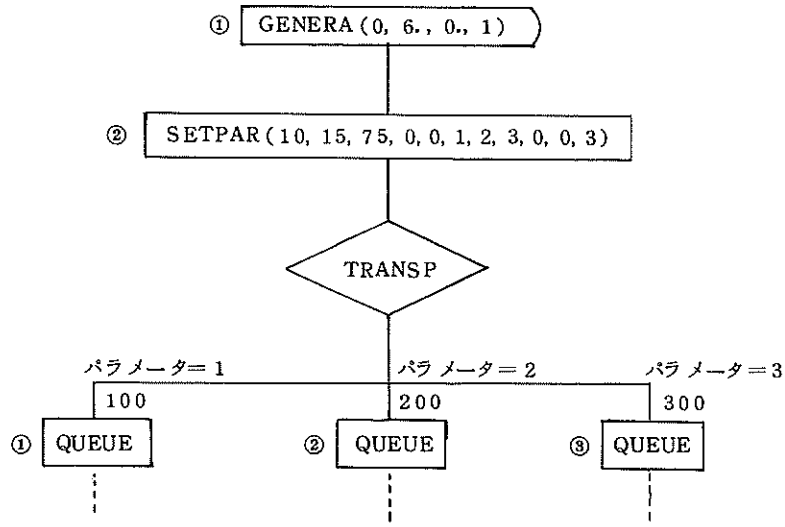
＊：呼IZCECのパラメータ番号Iのパラメータの値が3のときの飛先番地

＊：呼IZCECのパラメータ番号Iのパラメータの値が4のときの飛先番地

＊：呼IZCECのパラメータ番号Iのパラメータの値が5のときの飛先番地

〔仕様〕

通常SETPARでセットしたパラメータの値をみてその値によってフローを変えるのに用いる。



図－10

呼を発生させた後，各呼にパラメータを付与する。（例えば1は欧州航路船，2は北南米航路船，3はその他の定期航路船）これらの呼をパラメータの値によって分岐させる。（欧州航路船は1番の待ちに入る。北南米航路船は2番の待ち，その他は3番の待ちに入る。）

xiv) SUBROUTINE TRANSA

〔引数〕

I ：最初に探すFACILITY又はGROUP FACILITYのGROUP名LOGIC SWITCHのときはその番号

FN ：STORAGEかFACILITYかLOGIC SWITCHかGROUP FACILITYかを指示する。

＝1～STORAGE

＝2～FACILITY

＝3～LOGIC SWITCH

＝4～GROUP FACILITY

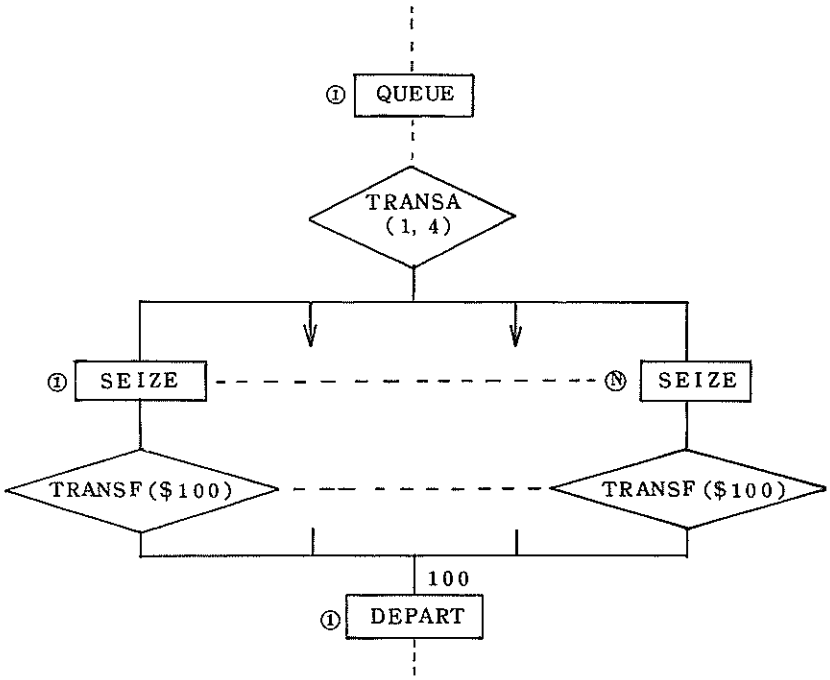
＊ ：$1011

NCS ：このサブルーチンの番号

メインのGO TO (……)，nのnと対応

〔仕様〕

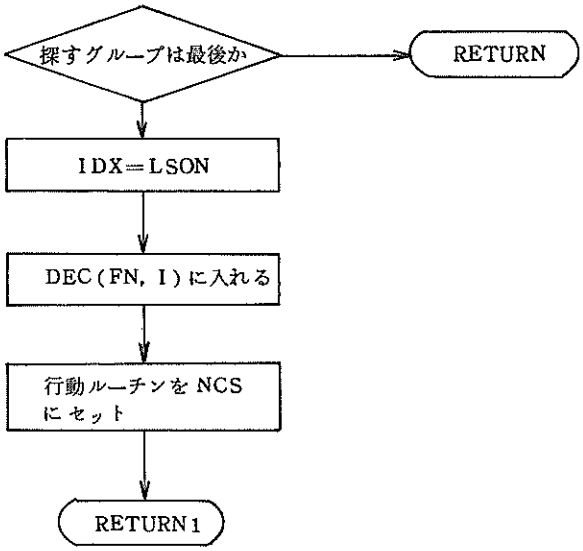
FACILITY STORAGE LOGIC SWITCH, GROUP FACILITY が使用不可のとき，その代替のものを探す必要のあるときに用いる。



図—11

FACILITY 1番からN番まで順次空いている もの　にとどまる。
を探し，いずれも空いていないときは 1番の QUEUE

〔フロー〕



XV) SUBROUTINE RESET

〔引数〕

ST ： 平均値

FN ： 分布型

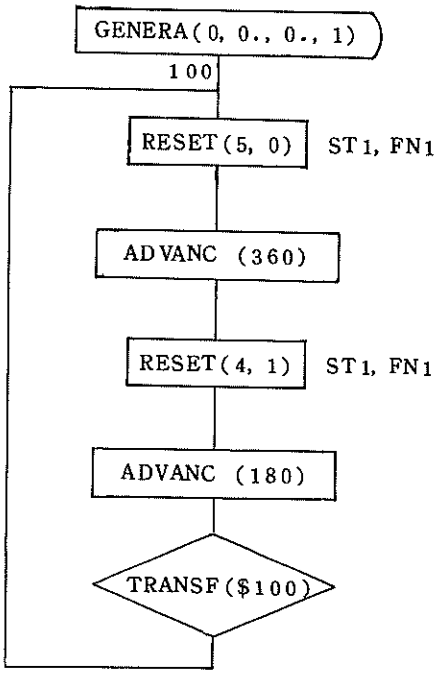
ST1 ： 変更すべき平均値

FN1 ： 変更すべき分布型

〔仕様〕

シミュレーションの時間帯によってサービス時間、もしくは到着時間間隔の分布型，平均値を変更したいときに用いる。

例



図－12

平均値5クロックの指数分布をST1, FN1 に与え，この状態を３６０クロック続けたのち平均値４クロックの一定分布に変更し，この状態を１８０クロック続けてもとにもどす。この操作をシミュレーションが終るまで繰り返す。

XVi) SUBROUTINE TRACE

〔引数〕

I ：TRACEをするかしないかを指示する。

＝1 ：TRACE しない。

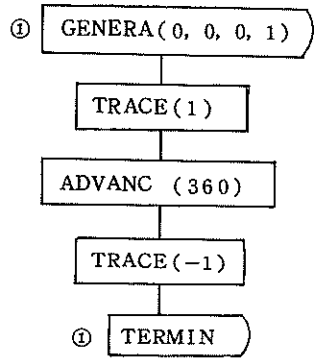
＝－1：TRACE する。

〔仕様〕

呼の行動を通過するサブルーチン毎に追跡したいときI＝－１を入れると呼の通ったサブルーチン名，時刻，その他の情報をラインプリンターに打ち出す。最初から全部TRACEしたいときはSIMULA の次に CALLする。

シミュレーション途中からTRACEしたいときはGENERA でコントロールして，次のように使うとよい。

例



図－13

シミュレーションが３６０クロック進んだときからTRACE を始める。

本ランでこれを使用すると時間がかかるので普通は，チエックランのとき等に利用する。

XVii) SUBROUTINE TERMIN

〔引数〕

NT ： TERMINの番号、普通GENERA の番号と一致させる。

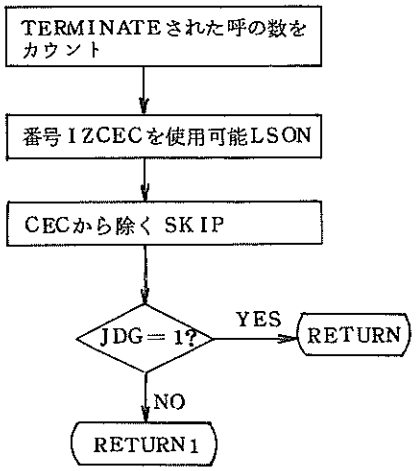
JDG： STARTと共に用いるとき１
それ以外は０を指定する。

＊ ： $1011

〔仕様〕

このルーチンに入った呼はシステムから除かれる。シミュレーションの終了時刻をコントロールするときはSTART と共に用いる。

〔フロー〕



xviii) SUBROUTINE SQUEEZ

〔引数〕

I　：FACILITY番号

K　：呼損があるか無いかを指定する。
＝0～呼損があるとき
＝1～FACILITYに先に割込みがかかっているとき，他の番地へ飛ぶとき
≒0 or 1～割り込みの為の待ちに入るとき

THIS：このチーチンの番号
メインのGO TO(……)，nのnと対応

＊　：TERMIN ルーチンのSTATEMENT番号

＊　：K＝1を指定したときの飛先番地
K＝1以外のときは$99999

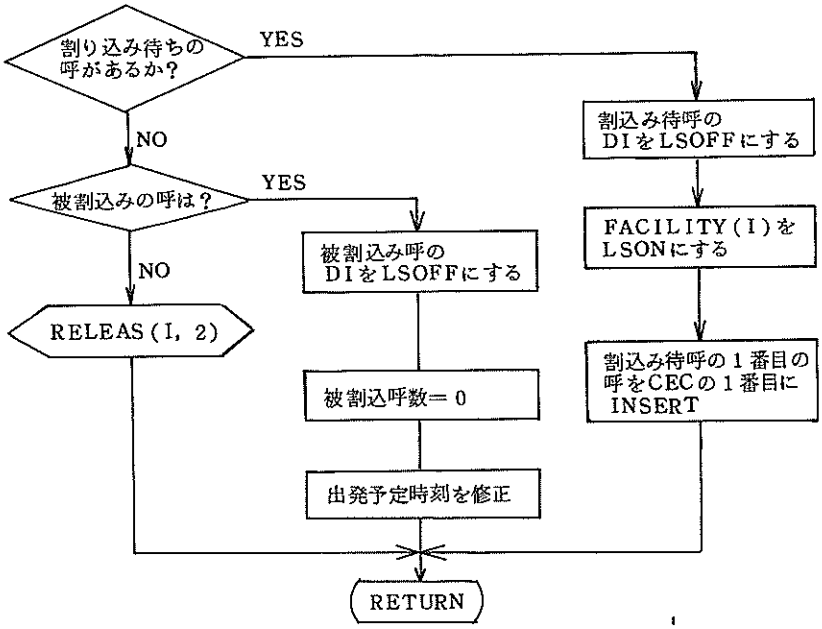
＊　：$1011

〔仕様〕

FACILITYが空いているかどうか判断し，空いていればそのFACILITYをSEIZEし，使用中であればそのFACILITYを割込み使用する。被割込の呼は割込んだ呼のサービスが終ると引きつづき残りのサービスを受ける。

次のRETURNと共に使用する。

〔フロー〕



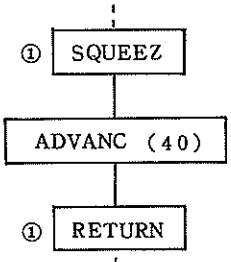
XIX) SUBROUTINE RETURN

〔引数〕

I　：割り込んだFACILITY番号

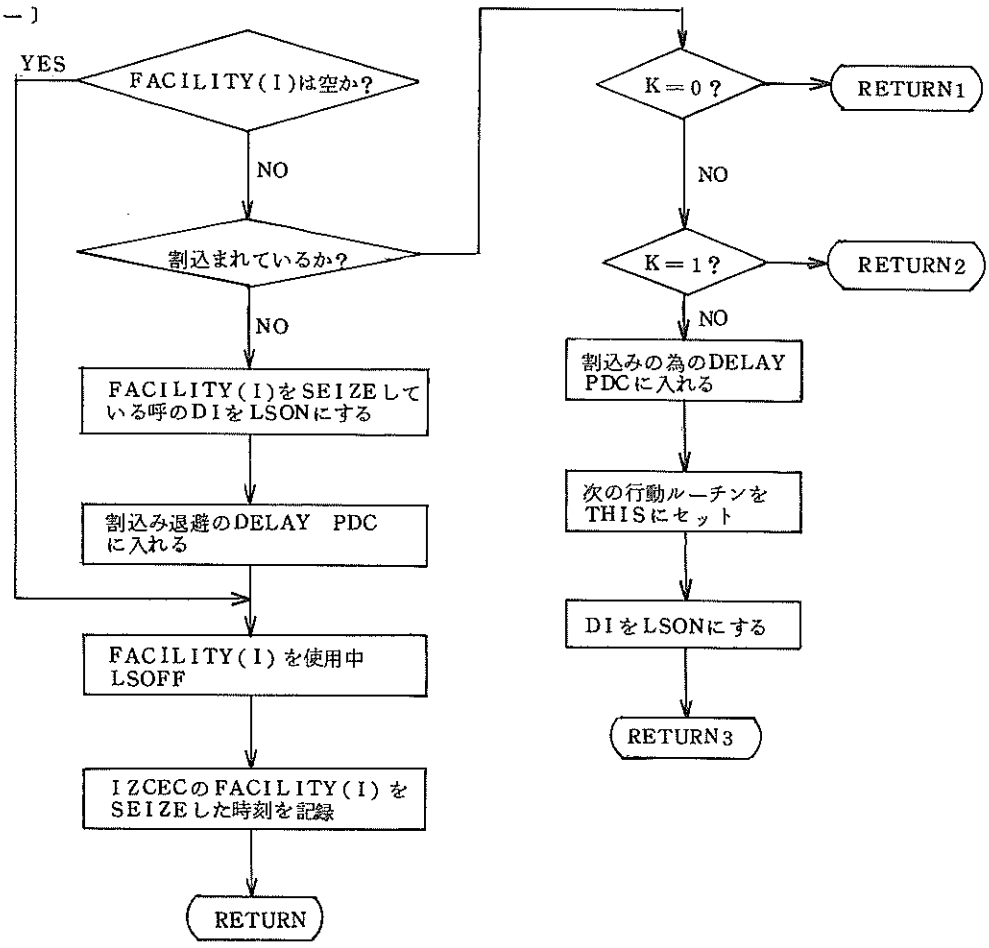
〔使用〕

SQUEEZと共に用い，割り込んだFACILITYを開放し，次の割り込みがあればその呼にFACILITYを開放なければ被割込み呼に開放する。



図－14

〔フロー〕



XX) SUBROUTINE STORAG

〔引数〕

N ：STORAGE の数

＊ ：CALL した次の STATEMENT 番号

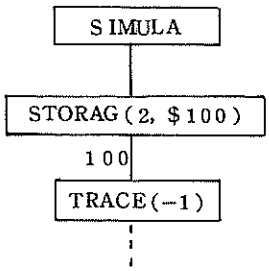
〔仕様〕

STORAGE の容量を定義する。

1～N番までのSTORAGEの最大容量を読み込む。

DATAは$DATAの後に(10I5)のFORMAT で指定する。

SIMULA の後，シミュレーションのCALL 系列の前に定義しなければならない。



図—15

XXi) SUBROUTINE PRIOF

〔引数〕

N：FACILITY番号

M：FACILITY番号

＊：次のSTATEMENT 番号

〔仕様〕

FACILITY が使用中であるとき順次代替できるものを探すときの代替の順序を定義する。

データは$DATAの後にカードで定義する。

FORMAT は(10I5)

例

```
CALL PRIOF(3, 3)
$DATA
    1  1  2  3
    2  2  1  3
    3  3  1  2
```

FACILITY1を最初に探すときは次に2，3，2を最初に探すときは次に1，3，3を最初に探すときは1，2を次に探す。

XXii) SUBROUTINE PRIOG

〔引数〕

N：グループの数

M：グループの数

＊：次のSTATEMENT 番号

〔仕様〕

FACILITYのグループに対してPRIOFと同じ定義をする。

XXiii) SUBROUTINE GRQUPM

〔引数〕

N：グループの数

M：1グループに含まれるFACILITY の最大数

＊：次の STATEMENT 番号

〔仕様〕

GSEIZEを用いFACILITY数個をグループとして取扱かうときこのサブルーチンでグループに含まれるFACILITY を定義する。

XXiv) SUBROUTINE ENTER

〔引数〕

I ： STORAGE の番号

J ： 占有容量

＊ ： $1011

NCS： このルーチンの番号

〔仕様〕

STORAGE が容量J分の空容量があるかどうか判断し空容量があれば，これを占有し，もし空容量がなければ待ちに入る。

STORAGEの最大容量はSTORAGで指定しなければならない。

次のLEAVEと対で用いる。

XXv) SUBROUTINE LEAVE

〔引数〕

I：占有していたSTORAGEの番号

J：占有していたSTORAGEの容量

〔仕様〕

ENTERと共に用い，占有していたSTORAGE を開放する。待ちがあればその待ちの中から使用しえる呼があれば前から順に開放する。

XXvi) SUBROUTINE SKIP

〔引数〕

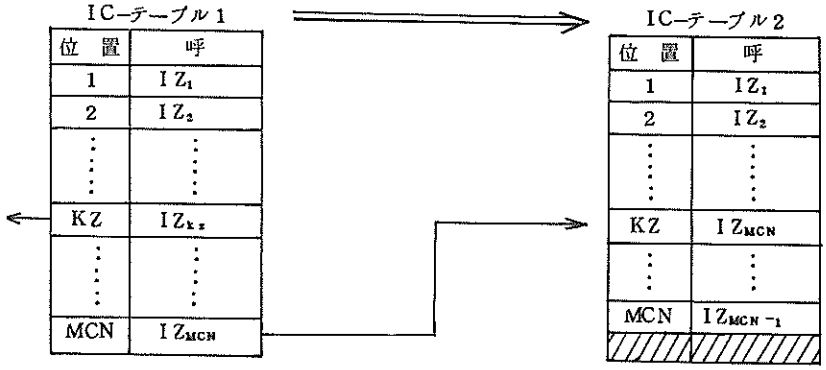
IC ： テーブル(表)名

KZ ： テーブル内の番地

MCN： テーブル内の呼の最大数

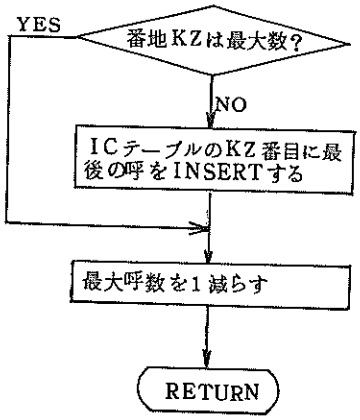
〔仕様〕

IC－テーブル(次図参照)のKZ番目の呼をDELETEし最後の呼をこの位置に INSERTする。通常USERは使わない。



図－16

〔フロー〕



XXVii) SUBROUTINE SPLIT

〔引数〕

IC : テーブル(表)名

MCN : テーブルに入っている最大呼数

I : テーブルに入れる呼の番号

〔仕様〕

IC-テーブルの最後に新たな呼をINSERTする。

通常USERは使用しない。

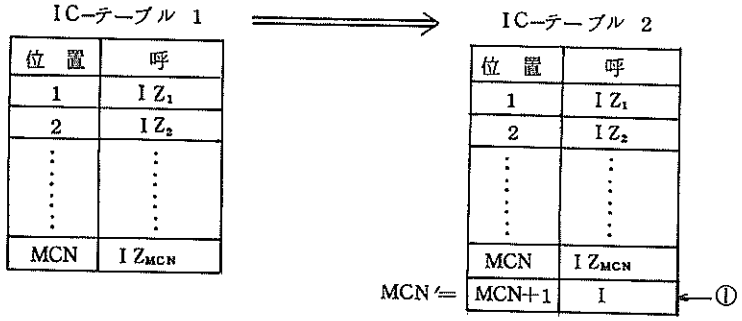


図-17

XXViii) SUBROUTINE INSERT

〔引数〕

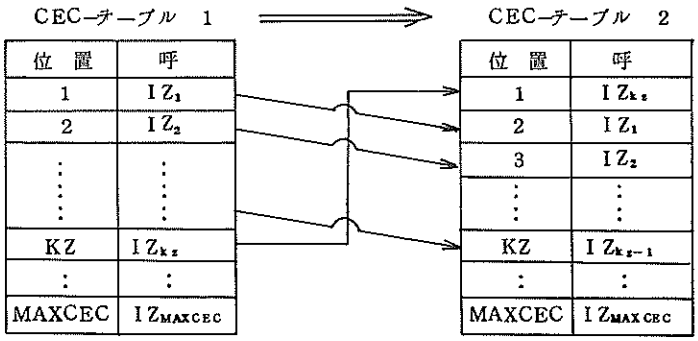
CEC : テーブル(表)名

MAXCEC : テーブル内の最大呼数

IZ : INSERTする呼の番号

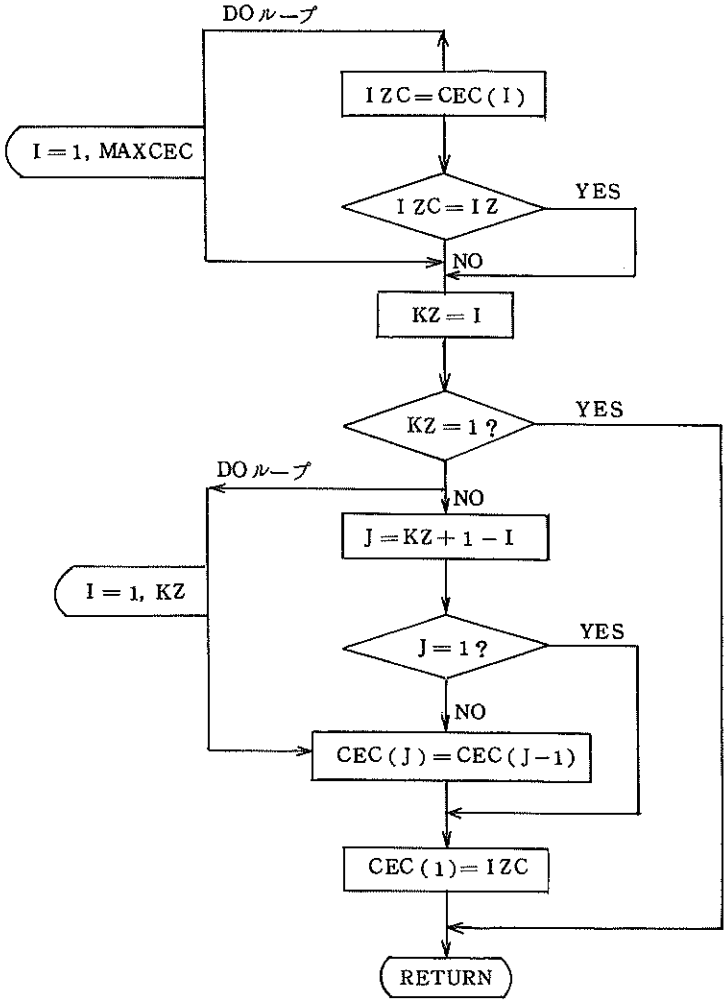
〔仕様〕

CECテーブルの呼をCECの1番目にINSERTし，テーブル内の呼の位置を1つづつ後へずらす。


CEC-テーブル 1
CEC-テーブル 2
位 置
呼
1
2
KZ
MAXCEC
位 置
呼
1
2
3
KZ
MAXCEC

図－18

〔フロー〕


DOループ
IZC＝CEC(I)
I＝1, MAXCEC
IZC＝IZ
YES
NO
KZ＝I
KZ＝1？
YES
DOループ
NO
J＝KZ＋1－I
I＝1, KZ
J＝1？
YES
NO
CEC(J)＝CEC(J－1)
CEC(1)＝IZC
RETURN

XXIX) SUBROUTINE DELAY

〔引数〕

SX ：FACILITY, STORAGE, LOGIC SWITCH等の区別

＝1～STORAGE

＝2～FACILITY

＝3～LOGIC SWITCH

＝4～その他

I ：STORAGE, FACILITY, LOGIC SWITCH等の番号

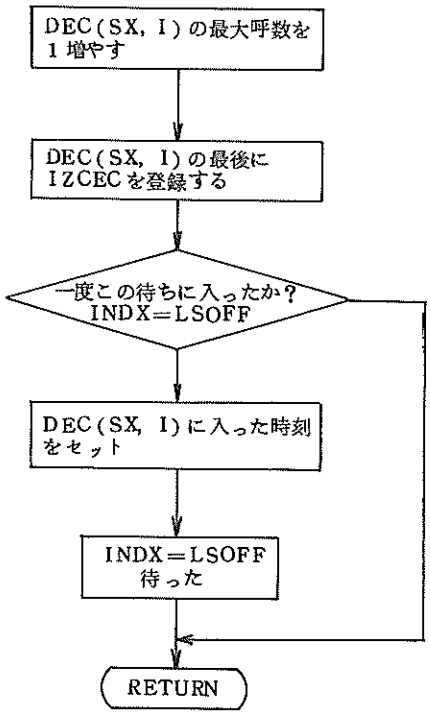
その他の場合は待ちの番号（DECの番号）

〔仕様〕

STORAGE, FACILITY, LOGIC SWITCH, GROUP FACILITY 等で使用中であったりSWITCHがLSOFFの状態であったときに，通過可能になるまでDECテーブルに登録しておく。このテーブルに入っている呼はQ・S・S・P のコントロールを受けない。

又，これはQUEUEにおける待ち時間算定の基礎になる。通常USERは使用しない。

〔フロー〕



XXX) FUNCTION IRAN

〔引数〕

A ：乱数の最大値

IR ：乱数初期値

〔仕様〕

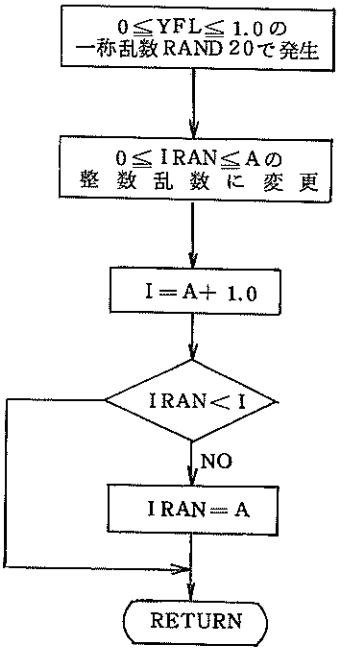
0＜x≦1の一称乱数を0＜I≦Aの整数一称乱数に変換する。

Aは実数値を与える。

IRは引数に直接数値を入れてはならない。

IRはなるべく大きな（5桁位）の奇数整数を入れる。

〔フロー〕



XXXI) SUBROUTINE RONE

〔引数〕

I ：DECの番号

FN ：DECの種類

＝1～STORACTE

＝2～FACILITY

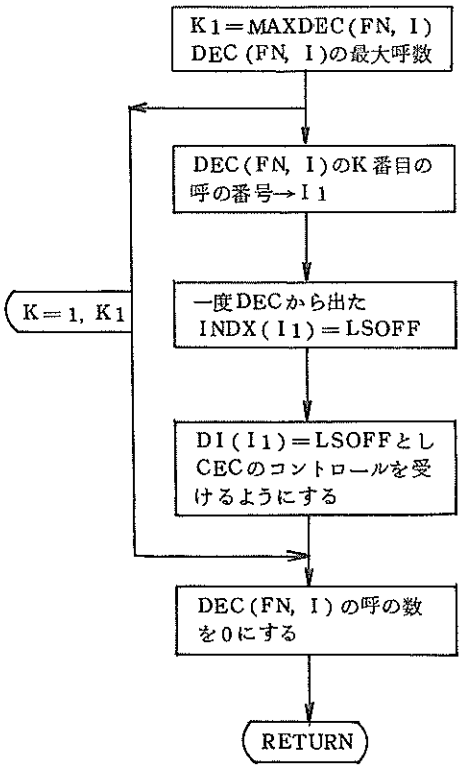
＝3～LOGIC SWITCH

＝4～その他

〔仕様〕

FACILITY STORAGE 等が使用可能になったとき，あるいは，LOGIC SWITCH が LSON の状態になったときにDECから呼を全部出しCECでのコント

ロールを受けられるようにする。
通常USERは使用しない。
〔フロー〕

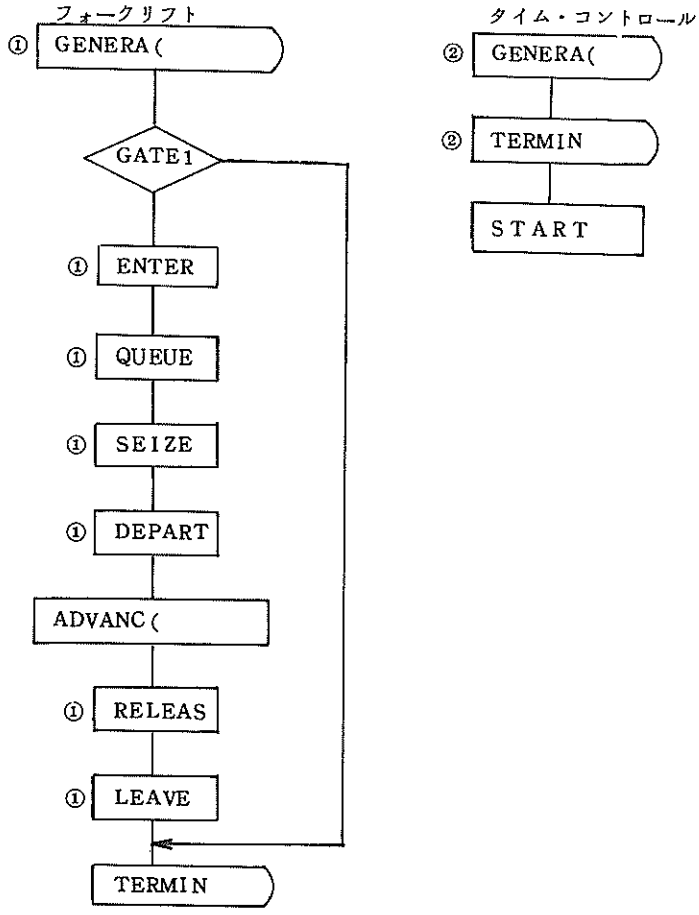


XXXii) SUBROUTINE RONE1
〔引数〕
I ：DECの番号
FN：DECの種類
＝1～STORAGE
＝2～FACILITY
＝3～LOGIC SWITCH
＝4～その他
〔仕様〕
FACILITY STORAGE 等が使用可能になったとき，あるいはLOGIC SWITCH が LSON の状態になったときにDECから呼を1番先頭のもの1つを出し，CECのコントロールを受けられるようにする。
通常USERは使用しない。

XXXiii) SUBROUTINE TRANSF
〔引数〕
＊：飛先番地のSTATEMENT番号
〔仕様〕
＊で指定した任意のSTATEMENT番号のルーチンに飛ばす。FORTRAN IVでのGO TO STATEMENTと同じ働らきをする。

7. プログラムの実例

〔例－1〕
クレーン1台のエプロンにフォークリストが平均10分の指数分布に従い到着し，荷役に30分かかる。このエプロン上にはフォークリストが5台まで駐車できるとし，このシミュレーションを8時間行なう。
(1) 与える条件としては，時間単位を分とし，
a．エプロンをSTORAGE 番号1，最大容量5とする。
b．クレーンをFACILITY 番号1とし，その前にできる待ち行列をQUEUE 番号1とする。
c．5台以上エプロン上で待っているときは他のクレーンの方へ逃げる（呼損）とする。
(2) このシミュレーションのフローチャートを図－19に示す。
(3) メインプロとコントロールカードをつけてコーディングしたものを図－20に示す。



図－１９

〔例－２〕

信号機のある横断歩道に車は平均２０秒の，人は平均４０秒の指数分布に従がい到着する。車は５０秒の青の間２秒かかって横断歩道を通過し，人は３０秒の青の間８秒かかって横断歩道を渡る。このシミュレーションを１時間行なう。

(1) 与える条件としては時間単位を秒にとり

a．横断歩道を容量１０００のSTORAGE１（人に対し）

b．車に対しては容量１０００のSTORAGE２ とする。

c．人の待ちに対しQUEUE番号１とする。

d．車の待ちに対しQUEUE番号２とする。

e．人に対して信号をLOGIC SWITCH1とし，

f．車に対して信号をLOGIC SWITCH2とする。

(2) このシミュレーションのフローチャートを図－２１に示す。

(3) メインプロとコントロールカードをつけてコーディングしたものを図－２２に示す。

```
TOSBAC-3400 TOPS-14 RELEASE 3 GENERATED 03/10/71

/EXC    FTC
                C
                C*************************************************************************
                C*                                                                       *
                C*        QUEUING SYSTEM SIMULATION PROGRAMMES                           *
                C*                              BY H.KURODA,,COMPUTATION CENTER          *
                C*                                                  22TH OF DEC, 1971    *
                C*                                                                       *
                C*************************************************************************
                C
                C              ********** COMMON VARIABLES **********
                C
      1               INTEGER  FEC, CEC, DI
      2               COMMON/MAIN1/NMAX,LSON,LSOFF,AT
      3               COMMON/MAIN2/IZFEC,FEC(200),MAXFEC
      4               COMMON/MAIN3/IZCEC,CEC(200),MAXCEC,KZCEC(200)
      5               COMMON/MAIN4/JJ(200),DI(200),TA(200)
                C
                C              ********** 1ST STEP OF QSSP CONTROL
      6                   CALL SIMULA(2)
      7                   CALL TRACE(*1)
      8                   CALL STORAG(1,¥1000)
      9          1000 IGCON=0
     10          1100 IGCON=IGCON+1
     11               IF(IGCON.GT.NMAX)GO TO 1001
     12                   GO TO(1,2),IGCON
                C
     13          1001 IZFEC=FEC(1)
     14               AT=TA(IZFEC)
     15               I=1
     16          1005 J=FEC(I)
     17               IF(TA(J).LT.AT)GO TO 1002
     18               GO TO 1003
     19          1002 IZFEC=J
     20               AT=TA(IZFEC)
     21          1003 I=I+1
     22               IF(I.GT.MAXFEC)GO TO 1004
     23               GO TO 1005
                C
                C              ****** SECOND STEP OF QSSP CONTROL
                C
     24          1004 I=1
     25          1008 J=FEC(I)
     26               IF(TA(IZFEC).EQ.TA(J))GO TO 1006
     27          1009 I=I+1
     28               IF(I.GT.MAXFEC)GO TO 1007
     29               GO TO 1008
     30          1006 IF(DI(J).EQ.LSON)GO TO 1009
     31                   CALL SPLIT(CEC,MAXCEC,J)
     32                   CALL SKIP(FEC,I,MAXFEC)
     33               DI(J)=LSOFF
     34               GO TO 1009
                C
                C              ****** THIRD STEP OF QSSP CONTROL


TOSBAC-3400 TOPS-14 RELEASE 3 GENERATED 03/10/71

                C
     35          1007 IF(MAXCEC.GT.0)GO TO 1010
     36               GO TO 1001
     37          1010 I=0
     38          1011 I=1
     39               IF(MAXCEC.LE.0)GO TO 1001
     40          1113 IF(I.GT.MAXCEC)GO TO 1001
     41               IZCEC=CEC(I)
     42               KZCEC(IZCEC)=I
     43               JZ=JJ(IZCEC)
     44               IF(DI(IZCEC).EQ.LSON)GO TO 1111
     45               GO TO 1112
     46          1111 I=I+1
     47               GO TO 1113
     48          1112 CONTINUE
     49                   GO TO(1,100,10,12,15,2),JZ
     50             1     CALL GENERA(1,0,3,,0,,0,¥1100,1)
     51           100     CALL GATE(1,1,1,¥1011,¥17,1,2)
     52            10     CALL ENTER(1,1,¥1011,3)
     53            11     CALL QUEUE(1)
     54            12     CALL SEIZE(1,4,0,0,¥1011,¥99999,¥99999)
     55            13     CALL DEPART(1)
     56            14     CALL ADVANC(4.,5,2,¥1011)
     57            15     CALL RELEAS(0,0,0)
     58            16     CALL LEAVE(1,1)
     59            17     CALL TERMIN(1,0,¥1011)
                C
     60             2     CALL GENERA(2,0,60.,0.,1,¥1100,6)
     61            20     CALL TERMIN(2,1,¥1011)
     62            21     CALL START(8,1,1,1,1,1,1,0,0,1,1,¥1011,¥99999)
                C
     63         99999 CONTINUE
     64          9999 STOP
     65               END
```

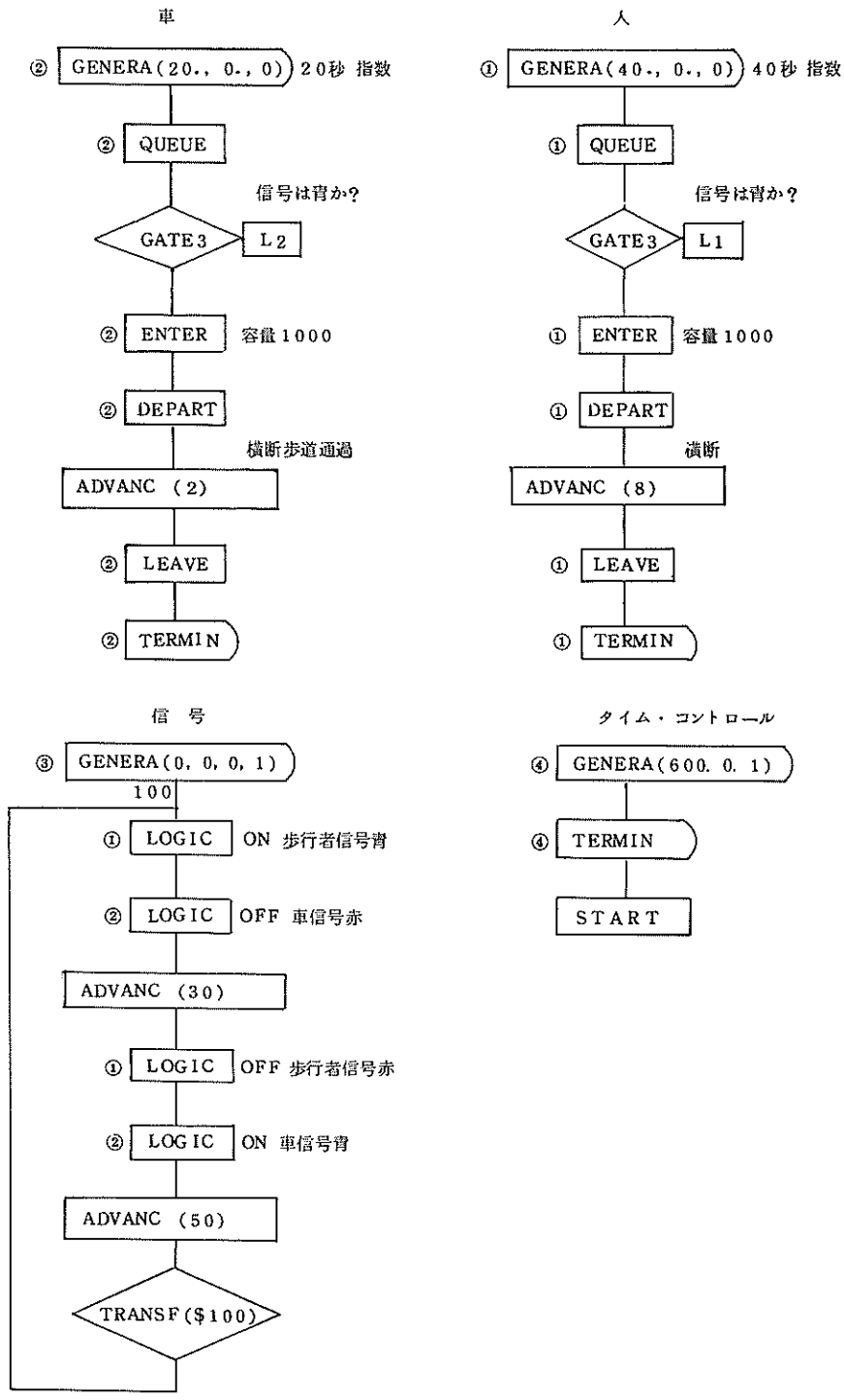
図－20－1

```
CLOCK TIME =      300

BLOCK NO,  GENERATED TRANSACTIONS  TERMINATED TRANSACTIONS
        1                   123                       119

** STATISTICAL PRINTOUT FOR STORAGES **
STORAGE            AVERAGE     AVERAGE           AVERAGE
NUMBER   CAPACITY  CONTENTS  UTILIZATION  ENTRIES  TIME/TRANS
     1          5      3580          716       73          14

** STATISTICAL PRINTOUT FOR FACILITIES **
FACILITY NO  AVERAGE      NUMBER   AVERAGE
             UTILIZATION  ENTRIES  TIME/TRANS
           1         979       73           4

** STATISTICAL PRINTOUT FOR QUEUES **
QUEUE NO   IAD     MAXQ     TWT
       1    74        2     780

HISTGRAM OF QUEUING TIME
          5         3         7        12        24        15         1         1         1         1
          0         0         0         0         0         0         0         0         0         0
          0         1         0         0         0         0         0         0         0         0
          1         0         0         0         1         0         0         0         0         0
          0         0         0         0         0         0         0         0         0         1
```

```
GATE                AT=  413.6  XACT=  5
ENTER        1      AT=  413.6  XACT=  5  CAPACITY=    1
QUEUE        1      AT=  413.6  XACT=  5
SEIZE        1      AT=  413.6  XACT=  5
DELAY**  2   1--    AT=  413.6  XACT=  5
GENERA       1      AT=  414.3  XACT=  1
GATE                AT=  414.3  XACT=  1
TERMIN       1      AT=  414.3  XACT=  1
GENERA       1      AT=  414.3  XACT=  3
GATE                AT=  414.3  XACT=  3
TERMIN       1      AT=  414.3  XACT=  3
GENERA       1      AT=  414.5  XACT=  1
GATE                AT=  414.5  XACT=  1
TERMIN       1      AT=  414.5  XACT=  1
RELEAS       1      AT=  414.9  XACT=  2
RONE1     1     2   AT=  414.9  XACT=  2
LEAVE        1      AT=  414.9  XACT=  2  CAPACITY=    0
TERMIN       1      AT=  414.9  XACT=  2
SEIZE        1      AT=  414.9  XACT=  5
DEPART       1      AT=  414.9  XACT=  5
ADVANC               AT=  414.9  XACT=  5
RELEAS       1      AT=  418.9  XACT=  5
LEAVE        1      AT=  418.9  XACT=  5  CAPACITY=    1
TERMIN       1      AT=  418.9  XACT=  5
GENERA       2      AT=  420.0  XACT=  4
TERMIN       2      AT=  420.0  XACT=  4
START
```

10%

図－20－2


車
② GENERA(20., 0., 0) 20秒 指数
② QUEUE
信号は青か?
GATE3
L2
② ENTER 容量1000
② DEPART
横断歩道通過
ADVANC (2)
② LEAVE
② TERMIN
人
① GENERA(40., 0., 0) 40秒 指数
① QUEUE
信号は青か?
GATE3
L1
① ENTER 容量1000
① DEPART
横断
ADVANC (8)
① LEAVE
① TERMIN
信 号
③ GENERA(0, 0, 0, 1)
100
① LOGIC ON 歩行者信号青
② LOGIC OFF 車信号赤
ADVANC (30)
① LOGIC OFF 歩行者信号赤
② LOGIC ON 車信号青
ADVANC (50)
TRANSF($100)
タイム・コントロール
④ GENERA(600. 0. 1)
④ TERMIN
START

図—21

```
          CALL SIMULA(4)
          CALL STORAG(1, $200)
  200     CALL STORAG(2, $1000)



          GO TO(1, 2, 3, 4), IGCON


          GO TO(1, 11, 12, 14, 2, 21, 22, 24, 3, 32, 35, 40), JZ
    1     CALL GENERA(1, 0, 20., 0., 0, $1011, 1)
   10     CALL QUEUE(2)
   11     CALL GATE(3, 2, 0, $1011, $101, 0, 2)
   12     CALL ENTER(2, 1, $1011, 3)
   13     CALL DEPART(2)
   14     CALL ADVANC(2., 4, 1, $1011)
   15     CALL LEAVE(2, 1)
  101     CALL TERMIN(1, 0, $1011)
C
    2     CALL GENERA(2, 0, 40., 0., 0, $1011, 5)
   20     CALL QUEUE(1)
   21     CALL GATE(3, 1, 0, $1011, $102, 0, 6)
   22     CALL ENTER(1, 1, $1011, 7)
   23     CALL DEPART(1)
   24     CALL ADVANC(8., 8, 1, $1011)
   25     CALL LEAVE(1, 1)
  102     CALL TERMIN(2, 0, $1011)
C
    3     CALL GENERA(3, 1, 0., 0., 1, $1011, 9)
  100     CALL LOGIC(1, 0)
   31     CALL LOGIC(2, 1)
   32     CALL ADVANC(30., 10, 1, $1011)
   33     CALL LOGIC(1, 1)
   34     CALL LOGIC(2, 0)
   35     CALL ADVANC(50., 11, 1, $1011)
   36     CALL TRANSF($100)
C
   40     CALL GENERA(4, 0, 600., 0., 1, $1011, 12)
   41     CALL TERMIN(4, 1, $1011)
   42     CALL START(4, 1, 1, 2, 0, 0, 1, 2, $1011, $9999)
```

図－22

8. あとがき

本プログラムは一応港湾・空港等交通システムのシミュレーションプログラムが組めるようなものを目指して作成したが，著者の貧弱なシミュレーションプログラミングの経験から作成したものであるので，今後種々の問題に適用する場合，本プログラム群だけでは記述しえないものが出てくる可能性は残されていると思われる。又，港湾技術研究所内の計算機（TOSBA−3400−41）を対照に作成した為，容量の制限（65Kword であるがユーザーの利用し得るのは多くても 45Kword 程度）からシミュレーションで得られる情報は最小限必要と思われるものにしぼられているし，オーバーレイ等の複雑な手法を用いなければならなかった為，ユーザーにとっては少々煩雑な手続きをとらねばならないので，システムシミュレーターとしてまだまだ改良の余地は残されている。しかしながら，従来のシミュレーションプログラミングに要した時間と費用に比べれば，はるかに進歩したものであろう。したがって今後，種々の問題に適用し，逐次修正追加し，より手軽なものと改良して行きたいと思う。

尚本資料作成にあたり，港湾技術研究所システム研究室工藤室長，計算室小川技官その他の御協力に対し感謝の意を表する次第であります。

（１９７１年１２月２７日受付）

参考文献

1）J. H. MIZE−J. G. COX共著　小笠原・青沼・秋葉・梅林・沢村共訳
：シミュレーションの基礎，倍風館

2）恵羅嘉男・岩田登志子・寺田脩共著
：システムシミュレーション，日刊工業新聞社

3）General Purpose Systm Simulator Ⅱ User's Manual, IBM

4）A. V. Gafarian and C. J. Ancker, Jr
：Mean Value Estimation From Digital Computer Simulation

港湾技研資料　No.133

1972・3

編集兼発行人　運輸省港湾技術研究所

発行所　運輸省港湾技術研究所
横須賀市長瀬3丁目1番1号

印刷所　日青工業株式会社

Published by the Port and Harbour Research Institute
Nagase, Yokosuka, Japan.