論理装置の故障診断*

北村拓郎** 稲垣正行**

1. まえがき

電子計算機，電子交換機等の論理装置が進歩発展するにつれて，高度の自動故障診断技術が要求されるようになった．その理由として，論理装置の機能の増大は装置の複雑化をもたらし，故障診断に高級な技術を必要とすると同時に故障を診断，修理する高度な保守技術者が不足するようになった．そのため従来の試験プログラムと人手による診断では故障修理の時間を短縮することが不可能になって来たことである．自動故障診断の目的は保全性 (Serviceability) を向上させることであり，尺度として MTTR (平均故障修理時間) の短縮にある．MTTR を短縮するために今までいろいろな故障診断方式が開発されてきたが，それらについて簡単にふれておく．詳しい内容については第3章で述べる．大規模な論理装置の有力な故障診断方式として，大きく3つに分類できる．そのひとつは No. 1 ESS 電子交換機に使用されている診断方式である．No. 1 ESS は実時間の電子交換機であるため，開発当初から高度の信頼性が要求されていたため保守方式，診断方式には相当の努力がはらわれている．電子交換機の故障診断の道具は診断プログラムであるが，修理時間を短縮するために診断辞書（保守辞書）を完備したところに独自性がある．辞書作成は実用的に確実性のあるハードウェア故障シミュレーターによってなされている．わが国では電電公社の電子交換機にこの方式が使用されている．電子計算機も最近，実時間的な利用が増加するにつれて診断方式が充実されてきた．とくに IBM 360 シリーズの大型機のために開発された診断方式として FLT (Fault Locating Test) は有名である．これは論理装置を非機能的な対象として取扱うもので対象論理装置の所定の入力に対する応答（出力）を故障がまったくない場合とすべての故障を仮定した場合の各々について求めておき故障したとき同じ入力に対する応答をみて故障箇所を指摘するものである．入力と応答の関係および診断辞書は DA (Design Automation) の情報を用いて自動的に作成できるため故障診断が確実となる．わが国では NEAC 2200/700 において実用化されている．IBM 360/30, 85 では FLT のかわりにマイクロ診断が使用されている．マイクロ診断はマイクロプログラムを用いて論理装置の診断を行なう方式である．以下，FLT，電子交換機の診断方式およびマイクロ診断に焦点をしぼって，論理装置の診断方式について述べることにする．

2. 論理装置とその故障[1)2)]

本論に入るまえに，重要な用語の定義と仮定について述べておこう．論理装置とは，離散的な有限値の信号を取扱う素子が相互接続されているような装置である．電子計算機，電子交換機，デジタル制御装置等が論理装置の一般的な例である．論理装置の解析と合成法はスイッチング理論の重要な一分野として開発されてきた．一般に論理装置は組合せ回路および順序回路の相互接続されたものと考えられる．組合せ回路は組合せ論理関数を実現するために使用されフィードバック線（閉ループ）の存在しないことによって特長づけられている．変数 $\{x_1, x_2, \cdots, x_n\}$ の集合の組合せ論理関数 y_i は現在の入力値によって決定される．すなわち $y_i = f_i(x_1, x_2, \cdots, x_n)$ である．組合せ論理関数は入力値の各組合せに関する関数値を示し，真理値表によって表現される．他に Karnough map によって表現することもできる．

順序論理関数は現在の入力値と過去の値によって決定される．そのような関数を実現する数学的モデルは通常，順序マシンまたは有限状態マシンとして取扱われる．その動作を実現する論理回路を順序回路と呼ぶ．過去の入力値は順序マシンの内部状態によって表現される．順序回路のモデルは Mealy[18] と Moore[19] の両者によって発表されている．Mealy マシンにおいて，任意の時間の出力はそのときの入力と内部状態によっ

* Fault Diagnosis of Digital System, by Takuo Kitamura and Masayuki Inagaki (EDP System-Engineering Division, Nippon Electric Co., Ltd.)

** 日本電気株式会社コンピュータ方式技術本部

てきまる．一方 Moore マシンでは出力は内部状態のみの関数によって表現される．時刻 t における入力，内部状態および出力を $x(t), s(t), z(t)$ とすると，次の状態 $s(t+1)$ を両モデルとも

$$s(t+1)=y(x(t), s(t))$$

によってあらわせる．

出力値 $z(t)$ は Mealy マシンにおいて

$$z(t)=z(x(t), s(t))$$

Moore マシンにおいて

$$z(t)=z'(s(t))$$

で表現することができる．任意の順序マシンはどちらか一方のモデルであらわされ，一つのモデルから他のモデルへの変換は常に可能である．

順序回路はクロックパルスによって制御されるかいなかによって，同期または非同期回路に分類される．一般的に同期回路はパルス入力によって特長づけられ，非同期回路はレベル入力とレベル出力によって特長づけられる．同期回路に関しては，入力は離散的にある特定の周期毎に生起すると仮定され，各入力パルスは回路の内部状態をおおくとも1回遷移する原因となる．このとき出力パルスはクロックパルスに同期している．組合せ回路と順序回路についての詳細は他の文献にゆずることにする．

つぎに本文で取扱う故障の型について述べる．故障は論理故障のみを取扱う．論理故障とは回路の論理的動作に変化を起こすものである．それゆえ，電圧，電流，パルスの波形，回路内の遅延に影響を与えるが，回路の論理関数に変化をきたさないような故障は考慮に入れないことにする．回路の動作不良は永久故障またはインターミテント（間欠）故障による可能性がある．インターミテント故障は実際には生起するけれども，それを試験する手順をもとめる論理はいまだ確立されていない．インターミテント故障は試験がおこなわれているとき，消失する可能性があるから，それを検出する確実な手段がない．本文では試験の実行中に，発生も消失もせず性質を変化しない永久（固定）故障のみを取扱うことにする．もし論理装置内にはいかなるときも一つの故障しか存在しないと仮定すると，テストパターンの発生は非常に簡単化される．単一故障の仮定はほとんどのテスト発生方法に使用されているが，試験と試験の間に一つ以上の故障が生起する確率が非常に小さくなるほど多数の試験が行なわれるならば，この仮定は十分正当性があるとみとめられる．数個の同時の論理故障を発生するような単一の物理的故障の生起確率もまた十分に小さくなければならない．しかしながら，それでも単一故障の仮定は回路の初期チェックのときにあきらかでない可能性がある．本文では単一故障の仮定を用いるものとする．

3. 主な故障診断方式

3.1 FLT (Fault Locating Test)[3)4)5)8)9)]

FLT は IBM システム 360 の自動診断方式として開発されたものであり，わが国では NEAC 2200/700 の診断方式として実用化されている．本章では両モデルの診断方式にもとづいて解説を行なう．

FLT による自動診断の原理は次のようなものである．自動診断は一連の刺激パターン（テストパターン）を被診断装置に挿入し，それに対する応答を観測することによってなされる．もしテストパターンと応答間の論理装置が一定の法則のもとに制御され試験されれば，各テストパターンに対する成功，不成功の一連の応答は故障回路素子の指摘を行なうために使用可能となる．FLT 試験方式を実現するために2つのシステムが必要である．一つはこのテストパターンと診断辞書をあらかじめ作成するソフトウェアシステムであり，他の一つはこのテストパターンを故障論理装置に挿入し，装置を診断するハードウェアと制御プログラムからなるシステムである．前者のシステムは FLT データ発生システムと呼ばれ，DA 論理マスターファイルにたくわえられている論理設計情報（論理ブロックに関する情報（パッケージ名，論理タイプ，回路図上の座標等），論理ブロックの接続情報等）を処理し，必要な情報を得るためのものである．詳しくは 4.1 で述べることにする．後者のシステムは FLT のためのテストパターンを挿入し，応答をとり出して，診断結果を解析し，以後の診断順序の決定等の機能を行うためのものである．これらの動作を制御するために，診断制御装置があり，被試験装置内のフリップフロップにテストパターンをセット (SCAN IN) したり，被試験装置のフリップフロップから応答パターンを観測 (SCAN OUT) したり，クロック制御 (CLOCK ADVANCE) を行なう．FLT 制御ルーチンは診断制御装置から FLT 試験を制御する．このとき FLT データ発生システムで得られた情報は入力媒体（磁気テープ等）に記憶されている．

IBM 360 の FLT 試験構成は図1のようになっている．試験は独立したハードウェアである FLT 制御部から挿入される．ロード信号が FLT 制御部に与え

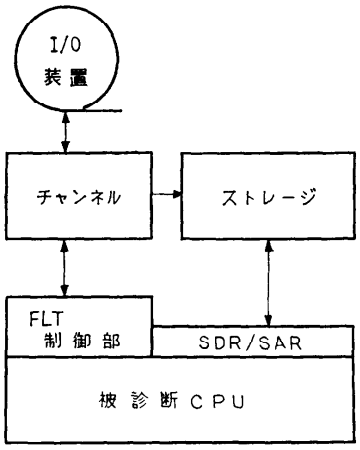

図 1 FLT 試験構成 (IBM 360)

られたとき，それはチャンネルと I/O 装置を動作させ，複数のテストパターンをストレージのバッファーへ読み込む．このパターンが読み込まれたとき，チャンネルは制御部へ信号を与え，SDR (Storage Data register) をとおして CPU へテストを与える．SDR は SAR (Storage Address Register) とともにハードコアテストであらかじめチエックされねばならない．FLT 制御部には，FLT 試験のルーチンがハードウェア的にセットされている．このルーチンのフローチャートを図2に示す．モデル 75 では FLT 制御部は論理的に CPU から分離されているが，物理的にはインテグレイトされている．モデル 50 と 65 では ROS (Read Only Storage) が FLT 試験のハードコアとして使用されている．故障箇所の指摘手順として，順序テスト法が用いられている．順序テスト法では一つの故障が初期テスト (Primary Test) によって検出されたならば，このテストは交換可能な故障パッケージを指摘するために十分な診断分解能を与えている可能性がある．もし初期テストが十分な分解能を与えているならば，FLT 試験は終了し，テスト番号が保守パネル上に表示される．FLT 用ドキュメント (診断辞書) には予想される故障パッケージがあらかじめ編集されている．もしその初期テストが十分な分解能を示さなければ，さらに FLT 試験が分解能をあげるために実行される．副次的なテスト (Secondary Test) は初期テストの結果が不一致であったパーティションの部分集合をチェックする．もしその部分集合がこのテストに対して不一致結果を示したならば，その部分集合が故障を含むことになる．もし成功すれば，故障はその部分集合の補集合に存在することになる．テストは図3に示されたテスト順序のように，終端点 (terminate) に到着するまで続行される．

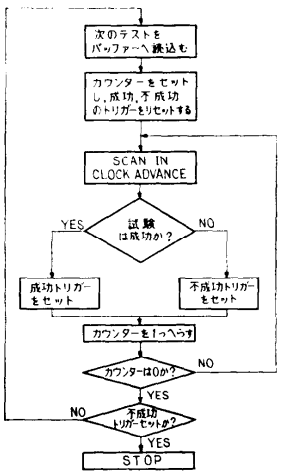

図 2 FLT 試験実行手順 (IBM 360)

NEAC 2200/700 の FLT 試験のシステム構成は図4のようになっている．FLT 試験に先立ってハードコア試験 (診断制御装置の機能試験)，データパス試験 (SCAN IN, OUT のデータパス，フリップフロップの試験) を保守計算機によって行なう．FLT 試験の実行手順は図5に示すように SCAN IN, CLOCK

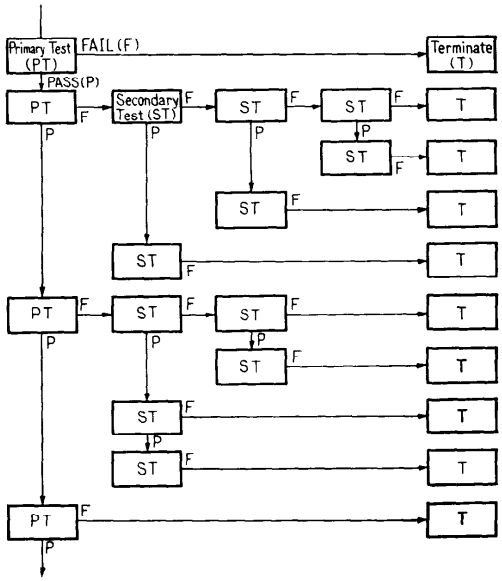

図 3 順序テスト法

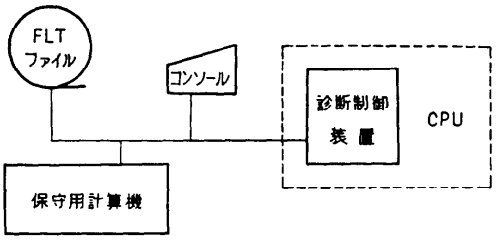

図 4 FLT 試験システム構成 (NEAC 2200/700)

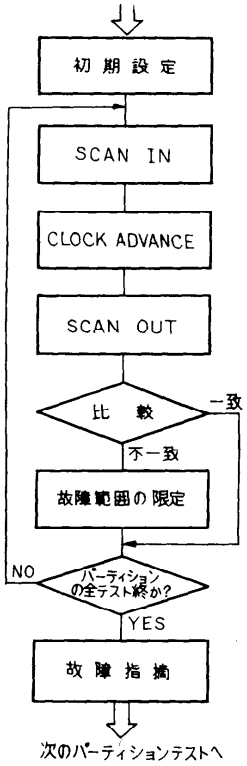

図 5 FLT 試験手順 (NEAC 2200/700)

ADVANCE, SCAN OUT, 比較, 故障範囲の限定, 故障指摘の順に実行される. 初期設定ではパーティション (出力フリップフロップからファンイン元のゲートを次々にたどって行き入力フリップフロップに至るまでの組合せ回路のこと) の制御情報をセットする. つぎにパーティションの入力フリップフロップにテストパターンをセットする (SCAN IN). SCAN IN が完了するとテストパターンによって論理装置を動作させるために正常のクロックを1ステップ進める (CLOCK ADVANCE). CLOCK ADVANCE の結果得られたパーティションの出力フリップフロップの

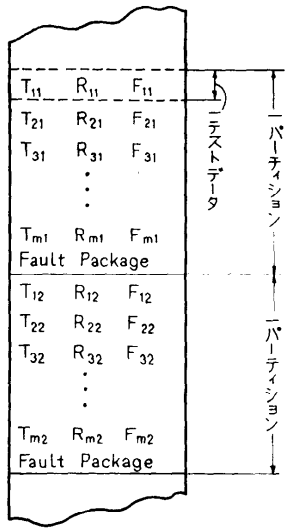

図 6 試験ファイル構造

値をとり出す (SCAN OUT). SCAN OUT によって得られた試験結果を期待出力値と比較する操作を比較という. 試験順序によって, 故障の範囲を限定することを故障範囲の限定という. 複数の試験の結果によって最終的に故障箇所の指摘を行なう. 故障箇所の指摘の手順として組合せテスト法を用いている. 組合せテスト法は故障テーブルを利用して故障を指摘する. 故障テーブルとはテストパターンがあるパーティション内のどの故障を検出するかを各テストパターン毎にテーブルとして示したもので, テストパターンと故障の1対1の対応表である. 故障が検出されたとき, 故障テーブルを参照し, そのパターンで検出される故障を疑しい故障とし, 各パターンで検出される疑しい故障との共通部分をとり, 故障の範囲を限定していく. パーティションへの全テスト終了後, 疑わしい故障としてしぼられた故障の属するパッケージを疑わしいパッケージとして指摘する. 診断用ファイルの構造は図6のようになっている. 下記に各々の構成要素を説明する. T_{ij} は SCAN IN データ, R_{ij} は T_{ij} に対する正しい出力結果, F_{ij} は T_{ij} で検出される故障 (故障テーブルの1部), 故障パッケージはパーティション単位に存在する故障Fに対する具体的な診断情報を各各示している. これらの T_{ij}, R_{ij}, F_{ij} で一つのテストデータを構成している. このファイルを用いて組合せテストを行なう手順は図7のようになる.

3.2 電子交換機の故障診断方式[10)11)]

ベル電話研究所で開発された No. 1 ESS 電子交換

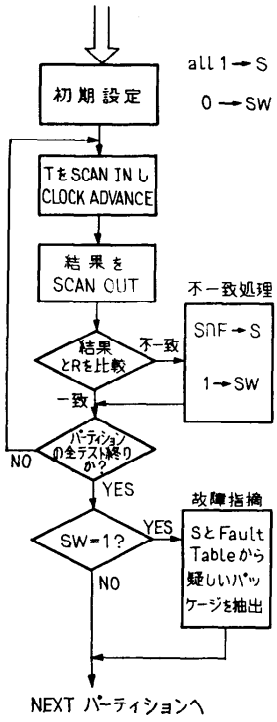

図 7 組合せテストの手順

機の診断方式にもとづいて説明する.(わが国の電電公社の DEX-1 の診断方式もほぼ同様手法を採用している. 詳細は本小特集号の他の解説において述べられているので, それを参照していただきたい.) 電子交換機は高度の信頼性を要求されるために中央制御装置(CC) を2重化した並列同期運転方式を用いて運用している. 特に CC の診断方式の概略は次のとおりである. CC のどちらか一方に故障が発生すると異常側を切離して正常側だけで呼処理を続行する. それから CC 故障診断プログラムが起動して正常側の CC は呼処理を行ないながら合間に異常側 CC の診断試験を行なう. 試験を行なった結果は見やすい形で編集されて, タイプライターに印字される. 保守員は印字内容を別に用意された診断辞書中に探索して故障箇所を知り, 交換可能な単位で交換する. 診断方式を定めるために考慮された点はつぎのとおりである. CC, 記憶装置とも2重化され, 正常時同期運転を行なう. 1命令実行のたびに両側 CC の記憶素子 (各種レジスター, フリップフロップ等) の内容を照合する機能を持つ. 二つの CC は互に相手側 CC の状態を読み取ることができ, また起動, 停止を制御する機能を持っている. 診断動作は呼処理を続けながら実行する必要がある. 診断の基本動作は特定の演算を CC に行わせ, その出力が正しいかどうか調べることである. 診断の目的のため CC に実行させる演算をテストとよび, プログラムによって行われる. 診断に際しては各回路ブロック毎に作られたテスト列をまとめて全部実行させる. それぞれのテスト実行の結果正しい出力が得られたことを成功, 正しくない結果が得られたことを不成功とすると, テストの成功, 不成功の組合せパターンからなる情報が得られる. これを診断出力と呼ぶ. 診断出力と故障位置との対応をつけるために診断辞書を作成する. CC は複雑な論理装置であるから, 人間による辞書作成が困難なため, 実際の CC に擬似的な故障を挿入してテストを実行させ, その結果を辞書の形に編集するのが実際的である. 擬似故障を作って辞書のデータを取る作業を能率化するために擬似故障発生装置を製作している. 擬似故障発生装置 (FSU) による辞書作成のシステム構成を図8に示す. 辞書作成手順は回路パッケージ情報を書込み可能記憶装置へロードすることからはじまる. その後で FSU をパッケージとそのソケット間に差し込み, ESS の中央制御装置に制御プログラムの実行を開始させる. これによって信号が中央パルス分配器を経て辞書制御装置 (DCU) へ行き, 最初の故障を FSU にセットする. それから ESS は診断プログラムを実行し, テスト結果が磁気テープに書込まれる. 被診断側の CC のすべての故

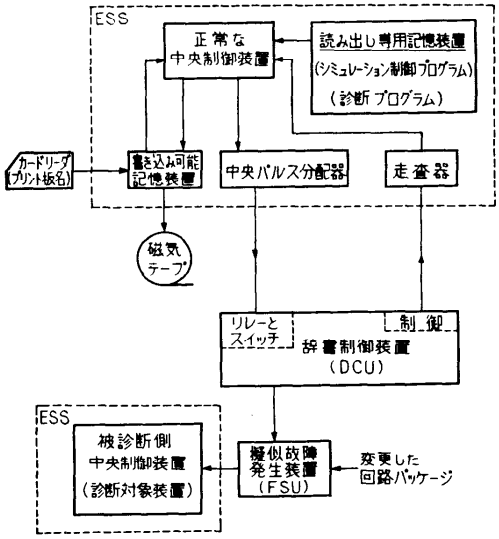

図 8 擬似故障発生装置による辞書作成システム構成 (No. 1 ESS)

障が挿入された後，汎用コンピューターによってシミュレーションデータが処理され，種々の診断辞書を作成する．

マイクロ診断のための辞書作成もほぼ同様の手順によって行われるが，それについては4.2で述べることにする．

3.3 マイクロ診断方式[12)]

マイクロプログラム制御の計算機はマイクロプログラムによる診断が可能となる．マイクロ診断方式を使用している代表的な計算機は IBM 360/85 と 30 である．本節ではモデル85の診断方式について解説する．

マイクロ診断とは定義されたハードウェアの一部分を特別に試験するために設計されたマイクロプログラムによる診断である．マイクロ診断には三つの部分がある．ROS (Read Only Storage) 内におかれた常駐診断 (Resident diagnosis) と WCS (Writable Control Storage) 内にある非常駐診断 (non-resident diagnosis) および非常駐診断を主メモリーへロードするために使用されるローダーである．ローダーは WCS において実行される．常駐マイクロ診断はすべてのデーターパスと初期プログラムのロードと WCS ルーチンのロードの実行に必要とするマイクロ命令の機能を試験するために使用される．非常駐マイクロ診断は残りの基本機能を試験するために使用される．まず最初にEユニット (Instruction Execution Unit) つぎにIユニット (Instruction Preparation Unit) そして最後に SCU (Storage Control Unit) の順に試験を行う．診断が終了するとコントロールは診断モニターにもどされ，通常のシステム 360 の診断が実行される．非常駐マイクロ診断プログラムはカード，テープまたはディスク等の I/O 装置から WCS にロードすることができる．マイクロ診断ローダーは非常駐マイクロ診断プログラムの最初の部分に存在する．それは全マイクロ診断を実行する間に非常駐マイクロ診断プログラムを主記憶にロードするために使用される．ロード手順は図9のフローに示される．

保守パネル上のロードマイクロ診断プシュボタン (LMD) をおすことによってマイクロ診断は初期値にセットされる．同時にコントロールは ROS 内の常駐マイクロ診断へわたされる．その結果，常駐マイクロ診断が実行される．もしこれが成功すれば，初期プログラムロード (IPL) ルーチンは I/O 装置から最初の非常駐プログラムを主記憶にロードする．この非常駐プログラムは最初のマイクロ診断とマイクロ診断用ローダーを含んでいる．IPL のおわりに最初の非常駐マイクロ診断が ROS 内のロード WCS ルーチンによって WCS へロードされ実行される．そしてローダーが最初のマイクロ診断をオーバーレイするように WCS にロードされる．次にローダーが実行され第2番目の非常駐マイクロ診断が I/O 装置から主記憶へはこばれる．ローダーはコントロールをロード WCS ルーチンへわたす．それは第2の非常駐マイクロ診断を WCS にロードしこのマイクロ診断を実行する．もしこれが最後のマイクロ診断でなければ，コントロールは再度ロード WCS ルーチンへわたされ，WCS にローダーを運ぶ．もし最後の非常駐マイクロ診断ならば，Iユニットをスタートさせ，コントロールを通常のシステム 360 の診断へわたす．以上のロード技術によって一連の常駐，非常駐，システム 360 の診断は単にロードマイクロ診断プシュボタンを押すことによって手動操作なしで実行することが可能となっている．

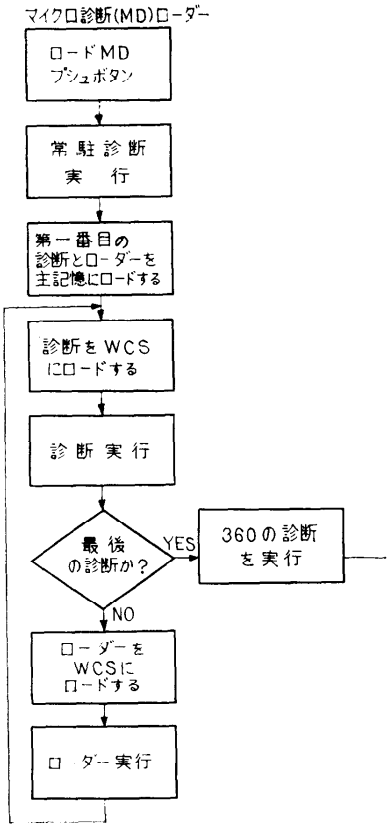

図 9 マイクロ診断手順

4. FLT とマイクロ診断の詳細

4.1 FLT データの発生[6][7]

3.1 で述べた FLT システムは非常に有力な診断方式であるが，それを有力ならしめるためにソフトウェアでサポートしているものに設計自動化システム (Design Automation System-DA) がある．DA は FLT で用いるテストパターンや故障指摘のための診断辞書，さらには保守診断のためのドキュメント作成情報などを発生させる資源となっている論理マスターファイルを提供する．これには論理装置におけるパッケージの実装位置やパッケージ間接続情報，または各パッケージ内での論理実装情報を含むものが用いられ，FLT パターン発生システムが固有の処理をほどこして目的のものを発生させる．図 10 に NEAC 2200/700 のための FLT パターン発生処理の概略フローを示す．

図 10 においてパッケージファイルとパッケージ接続ファイルとは DA で作成される論理実装情報が記述してあるファイルである．パッケージファイルにはパッケージ内の論理実装情報が，パッケージ接続ファイルにはパッケージの架内の実装位置ならびにパッケージ間接続情報が記述してある．

前処理はソースファイルであるパッケージファイルとパッケージ間接続ファイルからその回路記述および接続情報を FLT システムで取扱いやすいように論理ブロックの分割，論理ブロックの置換，接続情報，パッケージ毎のフリップフロップの順序付け，レベル割付け，クランプ信号トレースの処理を行い，ユニット内論理実装ファイルを作成する．

回路分割処理はユニット内をパーティションと呼ぶいくつもの組合せ回路に区分する．このパーティションが実際に FLT 試験を行う際のテストパターンをセットしたり観測したりする処理単位となる．出力ファイルであるパーティションファイルは分割されたパーティションの情報を含んでいる．

ドキュメント用編集処理ではパーティションファイルから各パーティションのために重要な情報（パッケージ，ブロック，コメントに関する情報）を抽出してユニット単位に出力する．ここで作られるパーティションインデックスファイルは保守診断の際のドキュメントとして，あるいは設計変更が生じた際の変更個所をさがす手段として用いられる．またこの処理で出力されるパーティション一覧表（パーティション名，ブロック数，入力フリップフロップ数，出力フリップフロップ数，論理段数，コメント情報等）はパターン発生以後の処理を円滑に行うための参考資料となる．

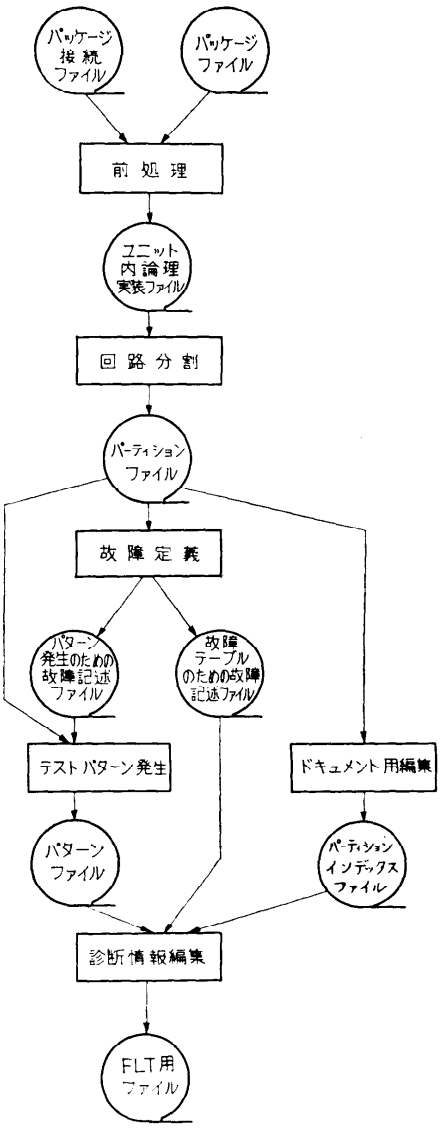

図 10 FLT パターン発生フロー

故障定義処理では回路分割処理で得られた各パーティションに対して故障定義を行う．故障定義の方法に二つあり，その一つはパターン発生時に挿入される故障定義であり，他の一つは故障診断に使用される故障テーブルのために定義される故障である．テストパターン発生のための故障定義は樹枝状回路（その中でフ

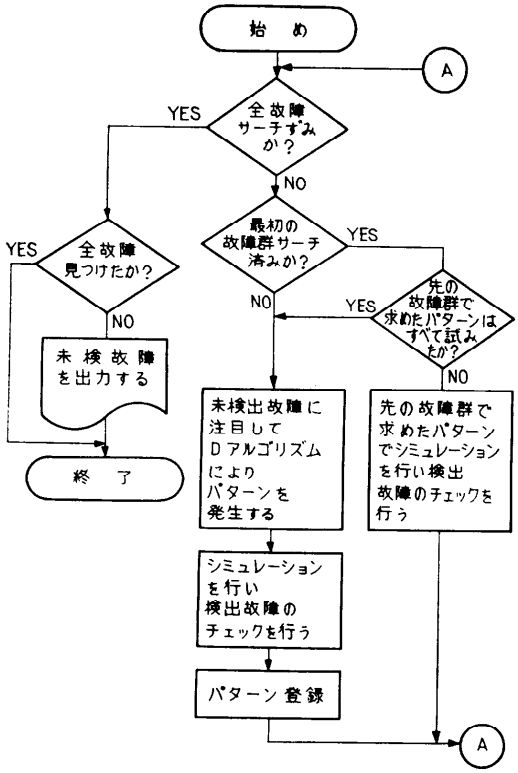

図 11 テストパターン発生手順

ァンアウト先に枝わかれのない組合せ回路）の入力となっている素子のピンに行い，故障診断のための故障定義は樹枝状回路の頂点（ファンアウトを二つ以上もつ素子）に行う．故障定義処理からパターン発生のための故障記述ファイルと故障テーブルのための故障記述ファイルが作成される．

パターン発生処理（図 11）は与えられた組合せ回路（パーティション）に対して定義した故障を検出するテストパターンの自動発生を行う．また，人手でテストパターンを発生し，それをパーティションの入力にセットして，シミュレーションを行い，試験パターンを登録する機能をかねそなえている．パターン発生用故障定義ファイルで定義された故障を検出するパターンを発生する方法としては J. P. Roth によるアルゴリズム[17]を採用している．また D-アルゴリズムにより求められた故障検出パターンでパーティション内の全故障シミュレーションを行う．まず，パーティション内の未検出故障の 1 つに注目して，D-アルゴリズムにより検出パターンを発生する．その際パターン発生の能率向上のため，パーティションの入力の論理故障に注目して発生を行う．求まった検出パターンは入力に Don't care な部分を持つが，その部分をランダムパターンで埋めてパーティション内の全故障の並列シミュレーションを試み，その検出パターンによって検出するすべての故障をチェックする．検出ずみの故障については D-アルゴリズムおよびシミュレーションの対象から除いていき，パターン発生の能率をあげている．パーティション内の全故障を一度にシミュレートすることが不可能な場合には全故障を適当な大きさのブロック（故障群）に分割して故障群単位のシミュレーションを行う．その際 2 番目以後の故障群についてパターン発生を行う場合にはあらかじめ発生されたパターンによりシミュレーションを行なって検出故障をチェックしておく．パターン発生の結果得られたテストパターンはパターンファイルにまとめられる．

診断情報編集処理は先のテストパターン発生処理において作成したテストパターンファイル，回路分割処理で作成したパーティションファイル，故障定義処理で作成した故障定義ファイル，ドキュメント用編集処理で作成したパーティションインデックスファイル等をもとにして，必要な情報を抽出し，それらを編集して試験時に必要な FLT ファイルを作成する．この処理のもう一つの機能として故障テーブルの作成と削減がある．故障テーブルというのは故障診断時に使用されるテーブルで，テストパターン発生によって得られたパターンをパーティションの入力フリップフロップにセットし，故障定義処理で作成された故障テーブルのための故障を挿入しながら並列シミュレーションを行なって得られる．故障の削減では他の故障によって包含される故障を除き，故障テーブルを簡単にする．

その他の処理として，未検出故障を抽出し，プリントアウトするルーチン，設計変更が起こった場合に使用する変更処理ルーチン等がある．

4.2 マイクロ診断と辞書作成[13][14][15]

FLT のように DA の情報を利用してテストデータ，診断辞書の自動作成が可能なシステムに関してはテストデータによる診断の確実性はある程度保証されている．しかしマイクロ診断を行なう場合にはあらかじめなんらかの方法でマイクロプログラムの診断可能性を評価し，故障指摘のための診断辞書を作成しておく必要がある．

マイクロ診断プログラム作成において注意すべき点は試験順序の決定とテストデータの作成である．試験順序の決定は通常 Start Small の思想で作成される

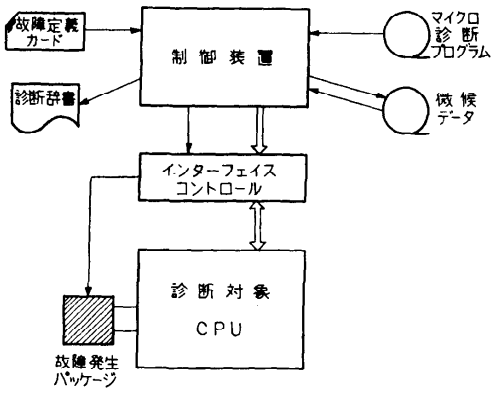

図 12 ハードウェアシミュレーション

これの試験手順は一般的に GO-NOGO の手法で行なう．またデータバス，演算回路系のマイクロ診断プログラムは，マイクロコンピュータの構造を解析して診断順序を決定することが可能である．これのマイクロ診断の試験手順は組合せテストによって行なわれる．テストデータは人手によって作成される．データバスやレジスター回路に関しては簡単なテストパターンを用い，演算回路系に関してはランダムパターン，真理値表解析などの方法を用いてテストパターンを得ることができる．しかし人手によるシミュレーションなので，故障検出，診断の確実性が保障されていない．そのためソフトウェアまたはハードウェアによるシミュレーションがぜひとも必要となる．同時に保守時の故障指摘のための診断辞書を作成する必要がある．

ハードウェアシミュレーション（図 12）とは論理装置のハードウェアモデルに実際に故障を挿入し，その動作を観察することによってシミュレーションデータがつくられる．これは 3.2 で述べた電子交換機の擬似故障発生装置による故障シミュレーションに相当する．ゆえにテスト群の完全性は正確に決定できシミュレーションデータも比較的信頼できる．シミュレーションの手順は故障発生，マイクロ診断（またはプログラム診断）の実行，診断結果の検出，診断辞書の作成となる．故障発生段階では DA 情報にしたがってパッケージに故障を発生させることができる擬似故障発生パッケージを準備する．制御装置からインターフェイスをとおして該当パッケージを制御することによって故障が発生される．つぎにマイクロ診断プログラムが外部記憶媒体からロードされ診断対象へ入力され，実行される．診断結果はインターフェイスコントロールを通して制御装置に伝えられ磁気テープに蓄積される．蓄積されたすべての診断結果を解析して診断辞書が作成される．

表 1 FLT とマイクロ診断の比較

比較項目＼診断方式	FLT	マイクロ診断
対象装置	一般論理装置（小中型計算機ではハードコアの比が大きくなるので大型機に適する．）	マイクロプログラム制御の論理装置（通常機能がマイクロプログラム制御のものが都合がよい．）
補助ハードコア（診断制御用ハードウェア）	FLT 試験を制御するハードウェアが必要となる（たとえば，診断用制御回路，SCAN IN OUT 用データパス等．）	マイクロプログラムがストアされる ROS（または WCS）を使用するから特に必要ない．
対象論理装置の論理設計	診断のための考慮は比較的少ない．	診断を行いやすいようにあらかじめ，回路構造の論理設計を行なう必要がある．
テストの発生	DA 情報を用いて自動的に発生する．	人手で発生する．
診断辞書	ソフトウェア故障シミュレーターにより作成する	ハードウェアまたはソフトウェアの故障シミュレーターにより作成する．
テスト・辞書の発生にかかる時間	テスト発生と診断辞書作成に時間がかかる．	マイクロ診断プログラムの作成にそれほど時間がかからないが，診断辞書作成に時間がかかる．
診断可能な範囲	SCAN IN，OUT するフリップ・フロップがカバー可能であるハードウェア．	マイクロプログラムが制御可能なハードウェア．
インターミテント故障の診断	それほど威力なし．	それほど威力なし．

ソフトウェアシミュレーションは通常与えられた初期値と入力に対して回路内のことなる線上の信号をシミュレートする論理シミュレーションである．ソフトウェアシミュレーションの応用はハードウェアシミュレーションと同様にすでに人手で作成されたマイクロ診断プログラムの診断可能性を評価することである．そのマイクロ診断プログラムで正常回路をシミュレーションしたあとで，特別な故障を挿入した回路がシミュレーション（故障論理シミュレーション）され，二つのシミュレーションの結果が比較される．これは特別な故障があたえられたマイクロ診断プログラムによって検出されたかをチェックするばかりでなく，故障が検出されたときの時間と出力値も計算することができる．この情報は診断辞書を作成するために使用される．このシミュレーションは与えられたマイクロ診断プログラムによって検出される故障集合を決定するためにも使用される．

4.3 FLT とマイクロ診断の比較

電子計算機の故障診断方式の代表として，FLT 方式とマイクロ診断の比較を一覧表（表 1）に示す．

5. おわりに

論理装置の故障診断について，現状で有力とされている診断方式を中心に解説し，とくに FLT とマイクロ診断の比較について述べた．将来ますます重要性を増すであろうこの分野について，すこしでも興味を持つ読者の参考となれば幸いである．

参考文献

1) Chang, H. Y., Manning, E. G. and Metze, G.: Fault Diagnosis in Digital Systems, Wiley-Interscience, 1970.
(鵜飼直哉，利谷圭介訳，ディジタル・システムの故障診断，産業図書)
2) Friedman, A. D., Menon, P. R.: Fault Detection in Digital Circuits, Prentice-Hall, Inc. 1971.
3) Carter, W. C. et al.: "Design of Serviceability Features for the IBM System/360," IBM Journal of Res. & Dev., Vol. 8, No. 2, pp. 115～126, 1964.
4) Clark, B. W.: "Automated Fault location on Processing units," Automated Support Systems Simposium for Advanced maintainability, 1967.
5) 北村，鳥居，若槻，梅野："NEAC 2200/700 における故障診断システムについて"，信学全大，No. 1151 (昭 46).
6) 北村，鳥居，舩津："NEAC 2200/700 における故障診断ソフトウェアシステムについて"，信学全大，No. 1153 (昭 46).
7) 北村，若槻，田中："NEAC 2200/700 における故障診断ソフトウェアシステム－パターン発生"，信学全大，No. 1154 (昭 46).
8) 北村，若槻，福井："NEAC 2200/700 における故障診断ソフトウェアシステム—診断情報の編集"，信学全大，No. 1155 (昭 47).
9) 北村，角田，若槻："NEAC 2200/700 における FLT コントロールプログラムについて"，信学全大，No. 1244 (昭 47).
10) Dowing, R. W., et al,: "No. 1 ESS Maintenance Plan," BSTJ., 43, pp. 1961～2019, 1964.
11) 徳山，本間："DEX-1 中央制御装置の障害診断"，通研実報，18, No. 6, pp. 1371～1386, 1969.
12) Bartow, E. and Mcguire, R.: "System 1360 model 85 microdiagnostics," SJCC Proceedings, 191～197, 1970.
13) Carrol, A. B., Kato, M., Koga, Y., and Naemura, K.: "A Method of diagnostic test generation," SJCC, pp. 221～228, 1969.
14) 稲垣，中井："マイクロ命令による動作試験，故障診断プログラムの作成"，信学会電算研資，EC 71-35 (1971).
15) Chang, H. Y., Thomis, W.: "Method of Interpreting Diagnostic Data for Locating Faults in Digital Machines," BSTJ, 46 (2), 289～317 (1967).
16) 北村，富田："RAS 技術の現状"，情報処理，Vol. 12, No. 8, 497～504 (1971).
17) Roth, J. P., et al.: "Programmed Algorithms to Compute Tests to Detect and Distinguish Between Failures in Logic Circuits," IEEE Trans. on Computers, Vol. EC-16, No. 5, 567～580, 1967.
18) Mealy, G. H.: "A Method for Synthesizing Sequential Circuits," BSTJ, 34, 1045～1080, 1955.
19) Moore, E. F.: "Gedanken-Experiments on Sequential Machines," Automata Studies, Princeton University Press, Princeton, New Jersey, 1956.

(昭和 47 年 6 月 21 日受付)