自動検針システム

仕様のモデル化とコード化

付属書２

目次

本書は、自動検針システムの仕様のモデル化及びそのコード化について述べたものである。

１．仕様のモデル化とコード化

仕様のモデル化の作業では、システム仕様書で定義されたシステムの静的な状態及び動的な振る舞いに、変数やその値、条件を割り当て、それらの相互の関係を状態遷移図で表現する。また仕様のコード化では、モデル化によって得られた状態遷移図を、モデル記述用の言語でコードに変換する。システム仕様のモデル化とコード化の手順を図１に示す。以下、それぞれの作業の結果を順に示す。

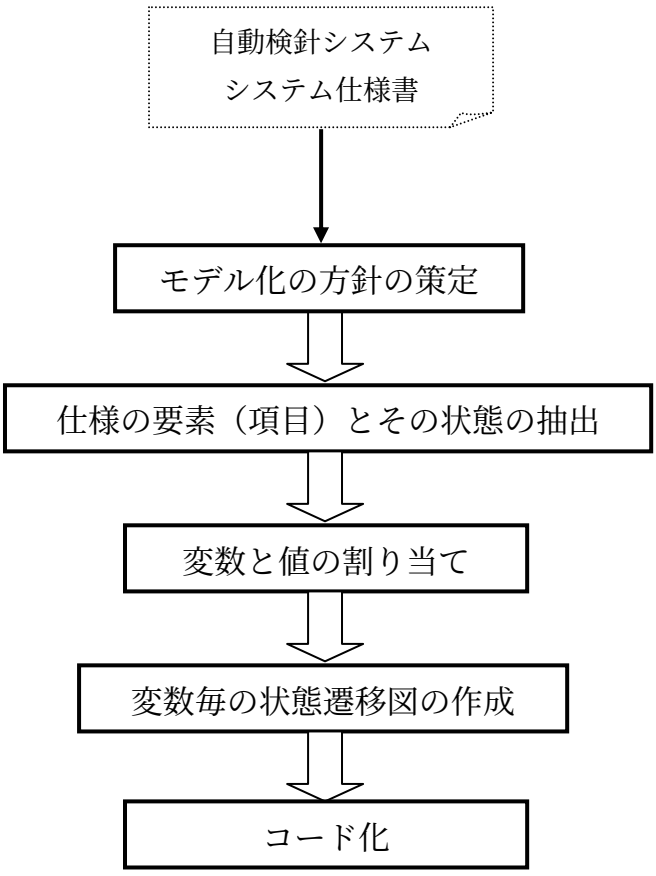


図１　システム仕様のモデル化とコード化

２．モデル化の方針の策定

自動検針システムの検証では、以下の２つの方針を立てた。

（１）設備の電源状態を記述する

自動検針システムの検証では、WHM内でのデータ設定と、センタ～GW～WHM間の通信に重点を置いているため、各設備の動作や設備間の通信に直接影響を及ぼす設備の電源状態に着目し、電源の状態変化がシステムに及ぼす影響をモデルに記述する。

（２）通信内容を抽象化する

設備間でやりとりする通信の内容（電文の構成、内容等）は、データ設定及び通信の手順に直接影響を及ぼさない。従って、モデルでは通信内容は可能な限り省略する。（ただし、GWから出力する指令データの正常／異常はモデルに記述する）

３．項目と状態の抽出

システム仕様書から抽出した項目とその項目の状態を表 1 及び表 2 に示す。

表１　項目とその項目の状態（1/2）

項目	状態
GWの電源	正常
	異常
WHMの電源	正常
	異常
GWとWHM間の通信	正常
	異常
センタからの要求	要求無し
	検針要求
	パターン設定要求１
	パターン設定要求２
	パターン設定要求３
	待機状態
GWからWHMへの指令データ	指令無し
	検針指令データ
	パターン設定指令１データ
	パターン設定指令２データ
	パターン設定指令３データ
	異常データ
WHMからGWへの返送	処理無し
	正常返送
	無応答
センタへの報告	報告無し
	完了
	失敗

表２　項目とその項目の状態（2/2）

項目	状態
パターン蓄積カウンタ	蓄積無し
	パターン１のみ蓄積
	パターン１＆２蓄積
	パターン２＆３蓄積
	全パターン蓄積
パターン設定完了フラグ	完了
	未完了
センタからの要求履歴	履歴無し
	パターン１の要求のみ出力済み
	パターン１＆２の要求を出力済み
	パターン１＆３の要求を出力済み
	全パターンの要求を出力済み
GWからの指令データ履歴	履歴無し
	パターン１の指令データのみ出力済み
	パターン１＆２の指令データを出力済み
	パターン１＆３の指令データを出力済み
	全パターンの指令データを出力済み
センタのリセットカウンタ	整数０～８

（１）設備の電源状態

自動検針システムを構成する設備（センタ、GW、WHM）の電源の状態は以下に示す通りとする。

●センタ：電源は常に、「正常」な状態とする。

●GW　：電源は任意に、「正常」あるいは「異常」のどちらかの状態を取り得るものとする。

●WHM：電源は任意に、「正常」あるいは「異常」のどちらかの状態を取り得るものとする。

センタの電源が異常の場合は、要求を出力することができずシステムとして全く動作しない状態となり、そのような状態は今回の検証では不必要であるため、センタの電源は常に正常として扱い、「センタの電源」は項目として定義しない。

（２）設備間の通信状態

「付属書１　自動検針システム　システム仕様書」では、設備間の通信状態は、入力側の設備と出力側の設備の電源状態に依存することとなっている。従って、各設備間の通信状態は以下のような項目として定義する。

（ａ）センタ～GW間の通信状態

センタの電源は常に正常であるので、センタ～GW間の通信状態はGWの電源状態と等しい。従って、センタ～GW間の通信状態の項目は、GWの電源状態の項目を用いることとする。

（ｂ）GWとWHM間の通信状態

「GWとWHM間の通信」を項目として定義する。また、電源状態と通信状態は順を追って変化する（例：電源が異常となった後に通信が異常となる）のではなく、同時に状態変化するものとして扱う。（モデルをコード化する際に、ＳＭＶのモデル記述用言語で用意されている、「ＤＥＦＩＮＥ文」を用いて実現する）

（３）センタからの要求

要求を何も出力していない「要求無し」の状態と、何らかの要求を出力している状態、要求を出力した後、GWからの報告を待っている「待機状態」を定義した。要求を出力している状態は１パルス（１遷移）とする。本項目の状態変化の順番を図２に示す。

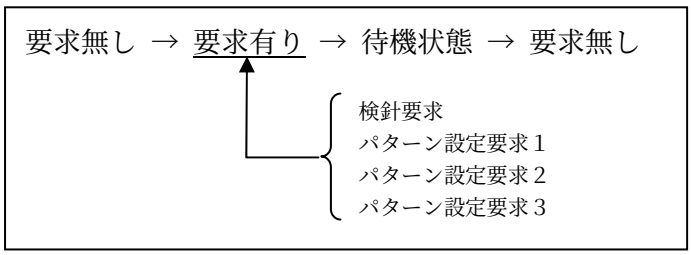


図２　項目：「センタからの要求」の状態変化の順番

（４）GWからWHMへの指令データ

指令を何も出力していない「指令無し」の状態と、何らかの指令を出力している状態、指令データが「異常データ」である状態を定義した。指令

を出力している状態は１パルス（１遷移）とする。本項目の状態変化の順番を図３に示す。

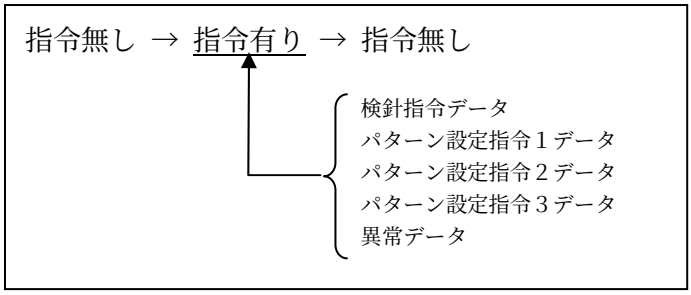


図３　項目：「ＧＷからＷＨＭへの指令データ」の状態変化の順番

（５）ＷＨＭからＧＷへの返送

ＧＷからの指令データを入力しておらず何も処理をしていない状態、すなわち初期状態に等しい状態を「処理無し」とした。正常な指令データを入力し返送を出力する状態を「正常返送」とした。また、ＧＷから指令データを入力したが、それが異常データであり、返送を出力できない状態を「無応答」とした。正常返送及び無応答の状態は１パルス（１遷移）とする。本項目の状態変化の順番を図４に示す。

処理無し →（正常返送、無応答）→ 処理無し

図４　項目：「ＷＨＭからＧＷへの返送」の状態変化の順番

（６）センタへの報告

報告を何も出力していない「報告無し」の状態と、処理が正常に完了したことを報告する「完了」の状態、処理が正常に完了しなかったことを報告する「失敗」の状態を定義した。報告を出力している状態は１パルス（１遷移）とする。本項目の状態変化の順番を図５に示す。

報告無し →（完了、失敗）→ 報告無し

図５　項目：「GWからWHMへの指令データ」の状態変化の順番

（７）パターン蓄積カウンタ

WHM内で、GWから入力したパターン設定指令（１～３）データを蓄積するメモリの変化を表現するため、「パターン蓄積カウンタ」を項目として定義した。本項目では、パターンデータを何も蓄積していない「蓄積無し」の状態と、何らかのパターンデータあるいは全てのパターンデータを蓄積している「パターン１のみ蓄積」～「全パターン蓄積」の状態を定義した。

（８）パターン設定完了フラグ

WHM内で、メモリ上に蓄積したパターンデータを本体に設定したことを示すフラグとして、「パターン設定完了フラグ」を項目として定義した。本項目では、パターン設定が完了した「完了」の状態と、パターン設定が完了していない「未完了」の状態を定義した。パターン設定が完了した状態は１パルス（１遷移）とする。本項目の状態変化の順番を図６に示す。

未完了 → 完了 → 未完了

図６　項目：「パターン設定完了フラグ」の状態変化の順番

（９）履歴

センタから要求を出力した際の履歴、GWから指令データを出力した際の履歴を表現するため、それぞれ、「センタからの要求履歴」及び「GWからの指令データ履歴」を項目として定義した。項目の状態はパターン蓄積カウンタと同様であり、表２の通りである。本項目はシステム全体を通してパターン設定の完了／未完了を検査する際、検査項目の条件として用いる。

（１０）センタのリセットカウンタ

GWから報告が出力されず、センタからの要求が「待機状態」でフリーズした場合に、それをリセットするタイミングをカウントするために本項目を定義した。カウンタが７となった時点で、センタからの要求は「要求無し」の状態に戻る。

４．変数と値の割り当て

抽出した項目と状態に、それぞれ変数と値を割り当てた。表 3 及び表 4 に項目と状態及びそれぞれに対応した変数と値の一覧を示す。

表３　項目と状態及びそれぞれに割り当てた変数と値（１／２）

項目	状態	変数	値
GWの電源	正常	GW_P	ON
	異常		OFF
WHMの電源	正常	WHM_P	ON
	異常		OFF
GWとWHM間の通信	正常	Com	1
	異常		0
センタからの要求	要求無し	Req	OFF
	検針要求		M
	パターン設定要求１		P1
	パターン設定要求２		P2
	パターン設定要求３		P3
	待機状態		Wt
GWからWHMへの指令データ	指令無し	Ord_Dt	EMP
	検針指令データ		M
	パターン設定指令１データ		P1
	パターン設定指令２データ		P2
	パターン設定指令３データ		P3
	異常データ		NG
WHMからGWへの返送	処理無し	Ret	EMP
	正常返送		OK
	無応答		NR
センタへの報告	報告無し	Rpt	EMP
	完了		OK
	失敗		NG

表４　項目と状態及びそれぞれに割り当てた変数と値（２／２）

項目	状態	変数	値
パターン蓄積 カウンタ 注）	蓄積無し	P_Cnt	EMP
	パターン１のみ蓄積		P100
	パターン１＆２蓄積		P120
	パターン１＆３蓄積		P103
	全パターン蓄積		ALL
パターン設定 完了フラグ	完了	P_Set	ON
	未完了		OFF
センタからの 要求履歴	履歴無し	Req_Hty	EMP
	パターン１の要求のみ出力済み		P100
	パターン１＆２の要求を出力済み		P120
	パターン１＆３の要求を出力済み		P103
	全パターンの要求を出力済み		ALL
GWからの 指令データ履歴	履歴無し	Dt_Hty	EMP
	パターン１の指令データのみ 出力済み		P100
	パターン１＆２の指令データを 出力済み		P120
	パターン１＆３の指令データを 出力済み		P103
	全パターンの指令データを 出力済み		ALL
センタの リセットカウンタ	整数０～８	Wt_Cnt	0 ～ 8

注）パターン蓄積カウンタの値は、図７に示すように割り当てる。

値	状態	P1 蓄積	P2 蓄積	P3 蓄積
EMP	蓄積無し	×	×	×
－	パターン１が未蓄積のため、 この状態は存在しない	×	×	○
－		×	○	×
－		×	○	○
P100	パターン１のみ蓄積	○	×	×
P120	パターン１＆２蓄積	○	×	○
P103	パターン１＆３蓄積	○	○	×
ALL	全て蓄積	○	○	○

○：蓄積有り　×：蓄積無し

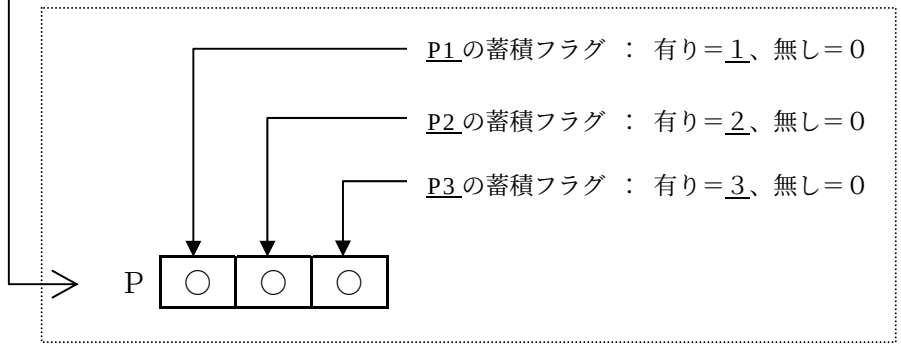


図７　パターン蓄積カウンタの値の割り当て

５．変数毎の状態遷移図

５．１　状態遷移図

それぞれの項目（変数）の状態遷移図を図 8 から図 19 に示す。

（１）WHMの電源（WHM_P）

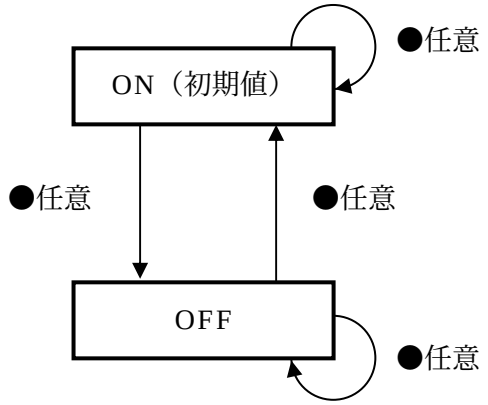


図 8　WHMの電源（WHM_P）の状態遷移図

（２）GWの電源（GW_P）

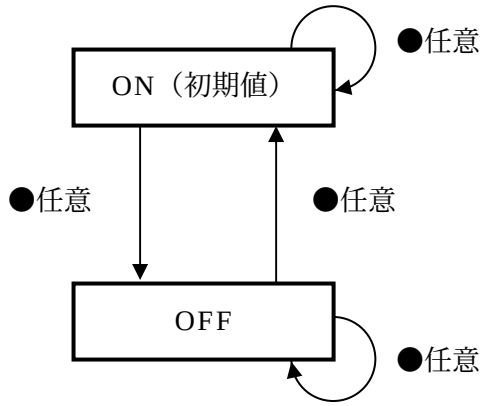


図 9　GWの電源（GW_P）の状態遷移図

（３）GWとWHM間の通信（Com）

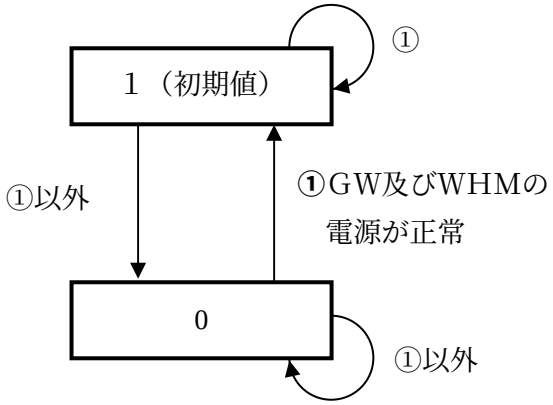


図 10　WHMとGW間の通信（Com）の状態遷移図

（４）センタからの要求（Req）

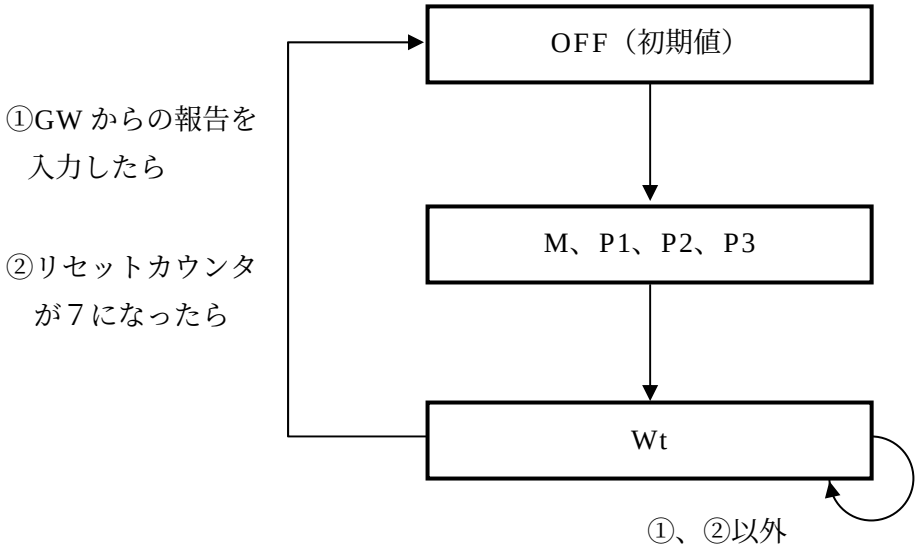
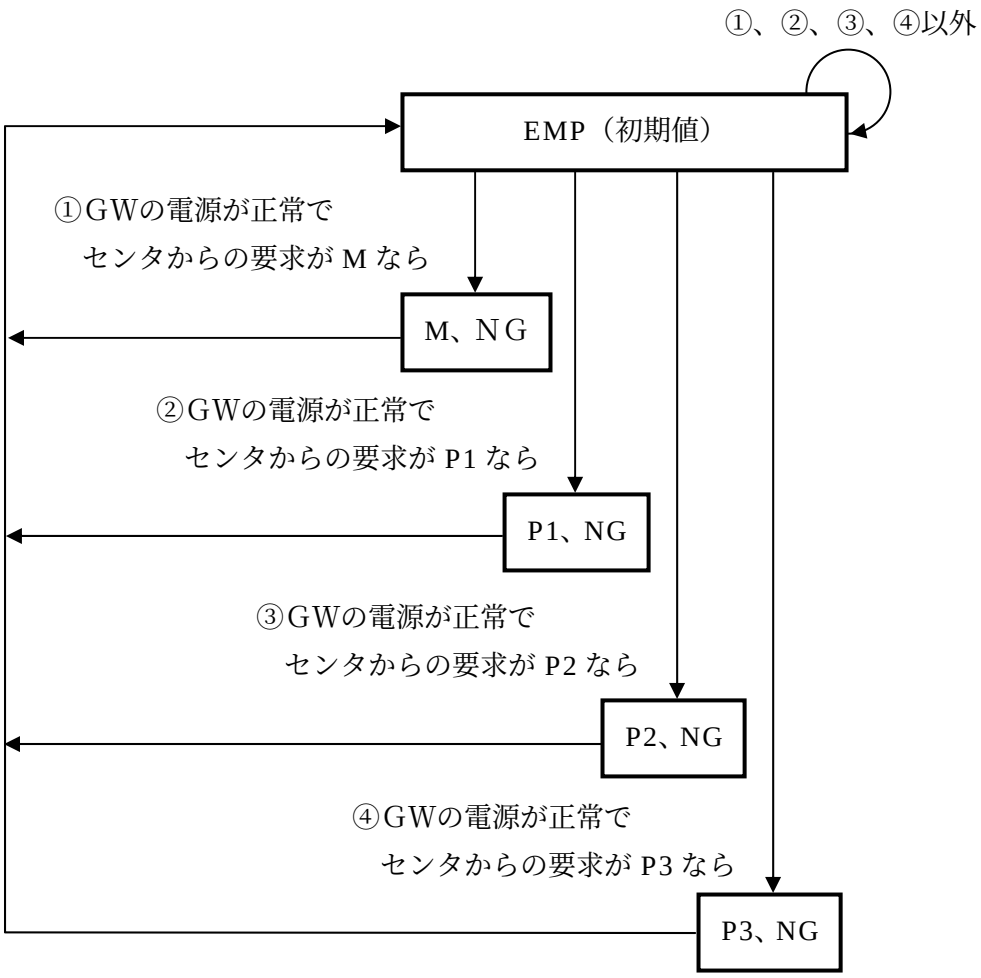


図 11　センタからの要求（Req）の状態遷移図

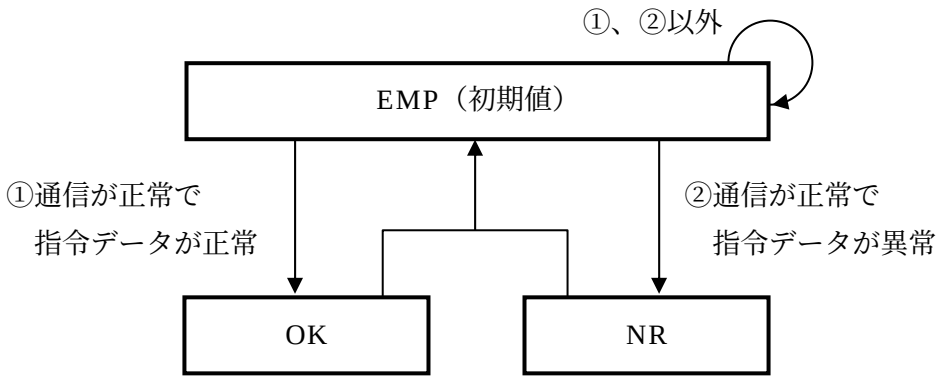
（５）GWからWHMへの指令データ（Ord_Dt）



注）M、P1、P2、P3 は正常データ、NG は異常データ

図 12　GWからWHMへの指令データ（Ord_Dt）の状態遷移図

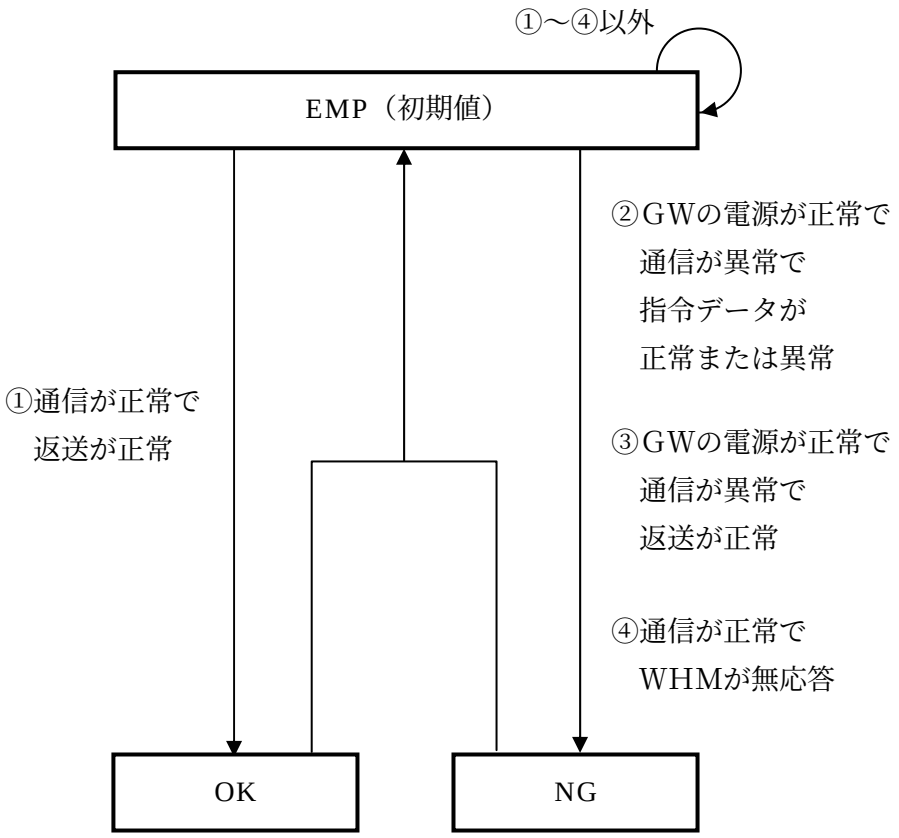
（６）ＷＨＭからＧＷへの返送（Ret）



注）②：通信は正常で、GWからWHMへの指令データの出力は完了したが、指令データが異常であった場合（WHM無応答の状態を表現している）

図 13　ＷＨＭから GW への返送（Ret）の状態遷移図

（７）センタへの報告（Rpt）



注）②：通信に異常が発生し、GWからWHMへの指令データの出力が正常に行えない場合

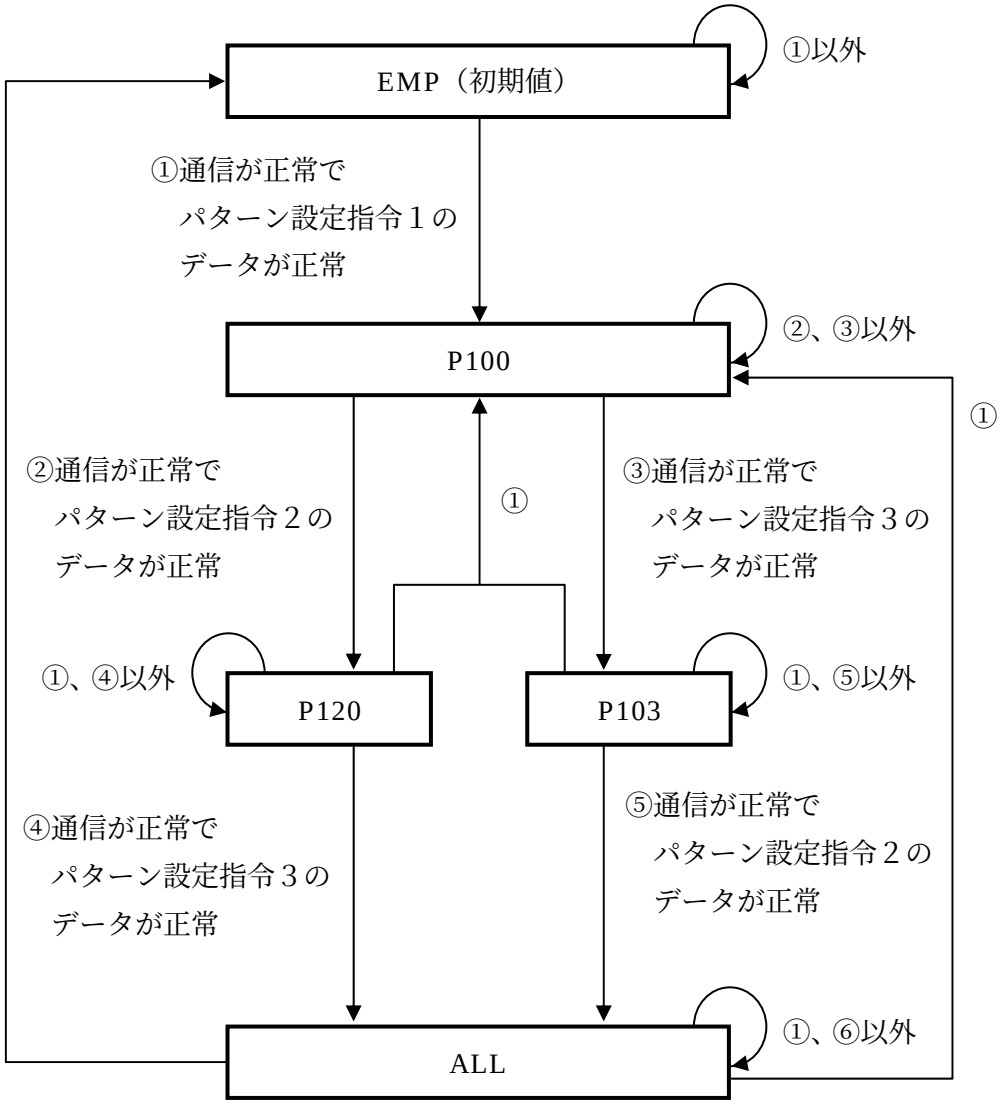
③：GWからWHMへの指令データの出力は完了したが、その後通信に異常が発生し、WHMからの返送が受け取れない場合

④：通信は正常で、GWからWHMへの指令データの出力は完了したが、指令データが異常でありWHMが無応答の場合

GWでは上記②、③、④の状態は区別できないので、全ての場合で「ＮＧ」の報告を行う。

図 14　センタへの報告（Rpt）の状態遷移図

（８）パターン蓄積カウンタ（P_Cnt）



⑥WHMの電源が正常で
パターン設定完了が ON で
①ではない

注）①：パターン設定指令１データを入力したらパターン蓄積カウンタは
P100 になる。
⑥：パターン設定が完了したらカウンタを 0FF（初期値）に戻す。

図 15　パターン蓄積カウンタ（P_Cnt）の状態遷移図

（９）パターン設定完了フラグ（P_Set）

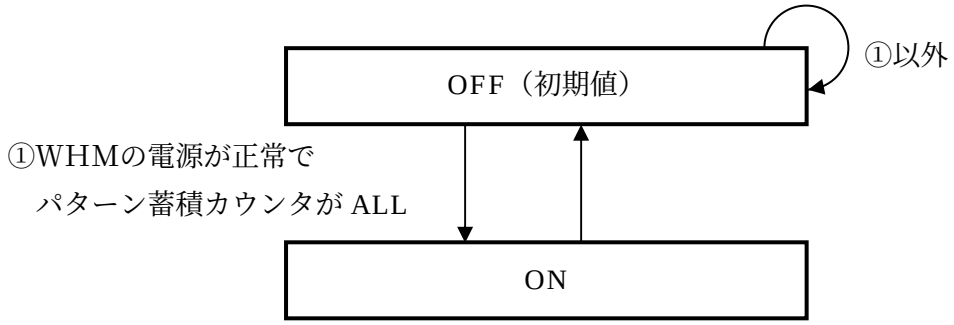
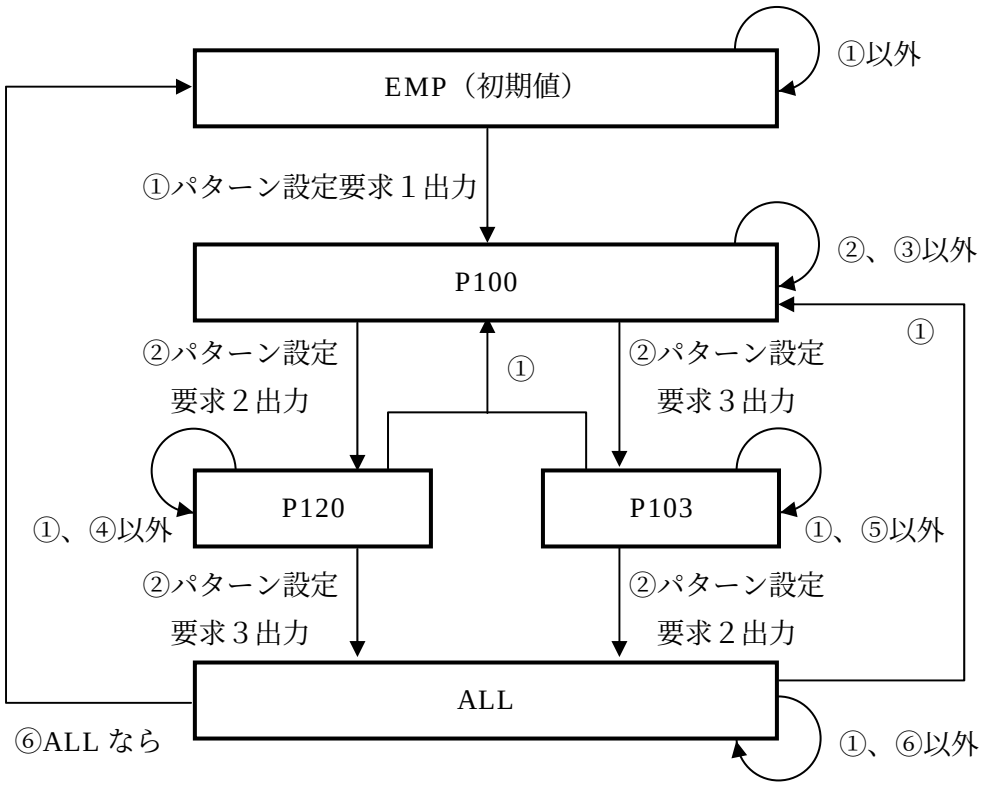


図 16　パターン設定完了フラグ（P_Set）の状態遷移図

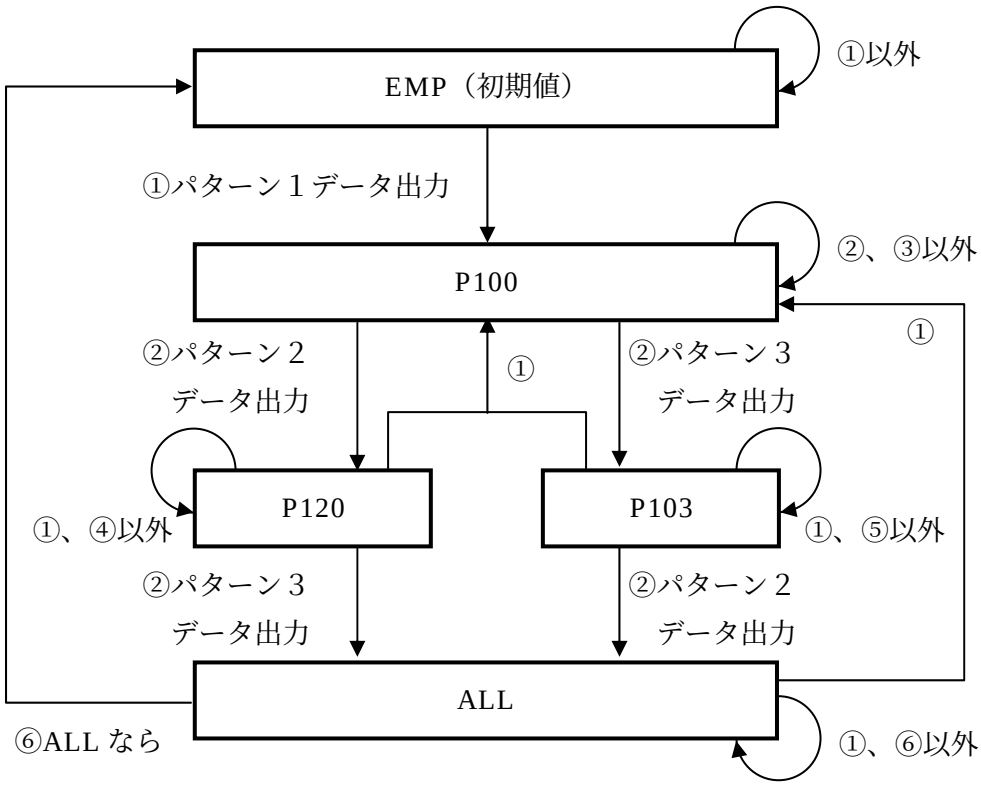
（１０）センタからの要求履歴（Req_Hty）



注）①：パターン設定要求１を出力したら履歴は P100 に戻す。
⑥：全パターンの設定要求を出力したら履歴を EMP（初期値）に戻す。

図 17　センタからの要求履歴（Req_Hty）の状態遷移図

（１１）GWからの指令データ履歴（Dt_Hty）



注）①：パターン１データを出力したら履歴は P100 に戻す。
⑥：全パターンのデータを出力したら履歴を EMP（初期値）に戻す。

図 18　GWからの指令データ履歴（Dt_Hty）の状態遷移図

（１２）センタのリセットカウンタ（Wt_Cnt）

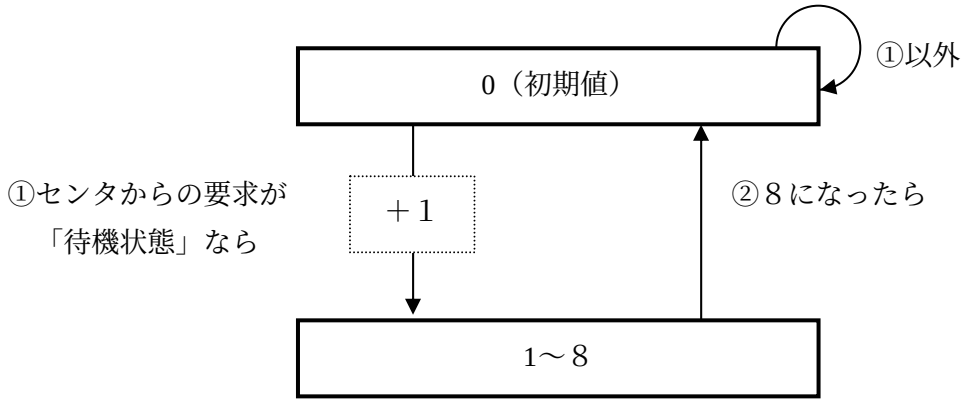


図 19　センタのリセットカウンタ（Wt_Cnt）の状態遷移図

５．２　状態変化順位

それぞれの項目（変数）の状態（値）が変化する際の優先順位を表 5 に示す。

表５　項目（変数）の状態（値）が変化する際の優先順位

順位	項目（変数）	変化する条件
1	GWの電源（GW_P） WHMの電源（WHM_P）	任意に変化する
	WHMとGW間の通信（Com）	GW_P 及び WHM_P と同時注）
2	センタからの要求（Req） ＧＷからＷＨＭへの指令データ（Ord_Dt） センタへの報告（Rpt） WHMからGWへの返送（Ret）	順位１及び同順位の項目の状態による
3	センタからの要求履歴（Req_Hty） GWからの指令データ履歴（Dt_Hty） センタのリセットカウンタ（Wt_Cnt） パターン蓄積カウンタ（P_Cnt ） パターン設定完了フラグ（P_Set）	順位１、２及び同順位の項目の状態による

注）WHMとGW間の通信（Com）は、GWの電源（GW_P）とWHMの電源（WHM_P）とを用いて「ＤＥＦＩＮＥ」にて定義されるので、それらと同時に変化する。

５．３　タイムチャート

それぞれの変数の値が変化する際のタイムチャートを以下に示す。（正常時のみ）

５．３．１　検針要求

センタからの検針要求に応じてGWからWHMへ検針指令データを出力し、その応答としてWHMから返送を入力し、結果をセンタへ報告する際のタイムチャートを図 20 に示す。

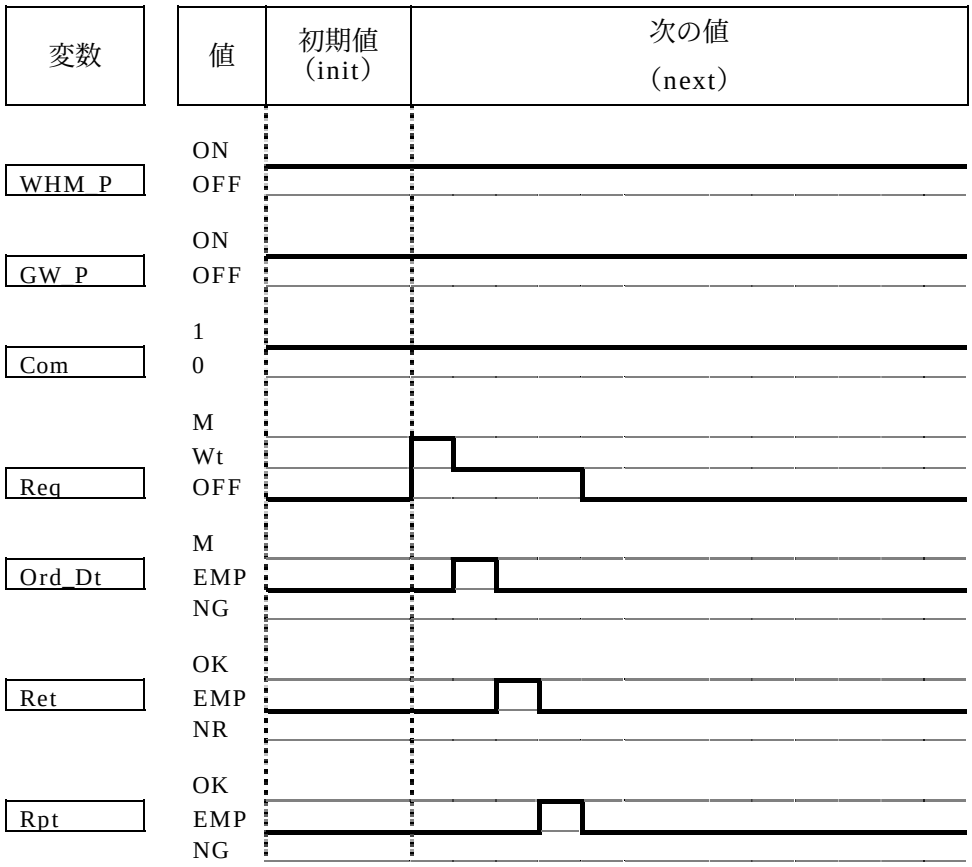


図 20　検針要求時のタイムチャート

５．３．２　パターン設定要求

センタからのパターン設定要求に応じてGWからWHMへパターン設定指令データを出力し、その応答としてWHMから返送を入力し、結果をセンタへ報告する際のタイムチャートを図 21 に示す。本図ではWHM内のパターン設定カウンタ及びパターン設定完了フラグのタイムチャートも同時に示す。

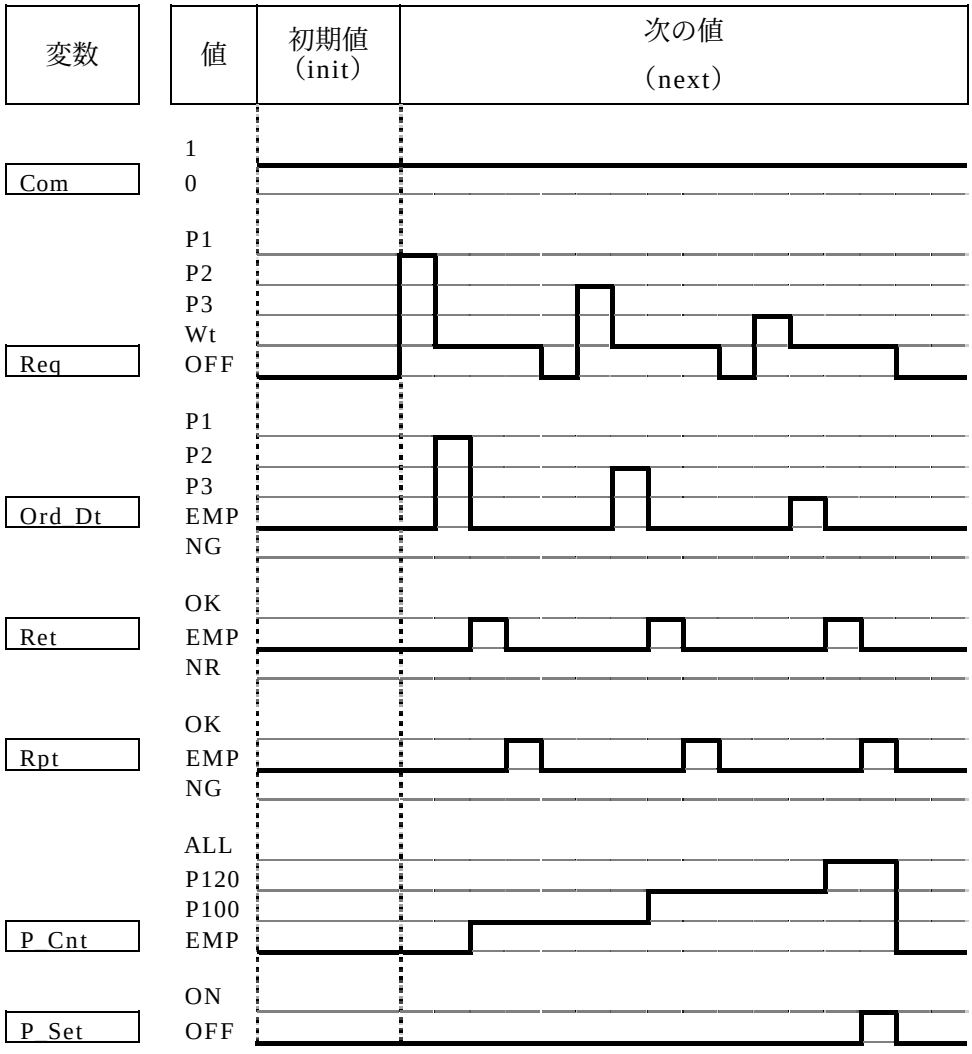


図 21　パターン設定要求時のタイムチャート

５．３．３　センタからの要求履歴及びＧＷからの指令データ履歴

センタからの要求履歴及びGWからの指令データ履歴のタイムチャートを図 22 に示す。

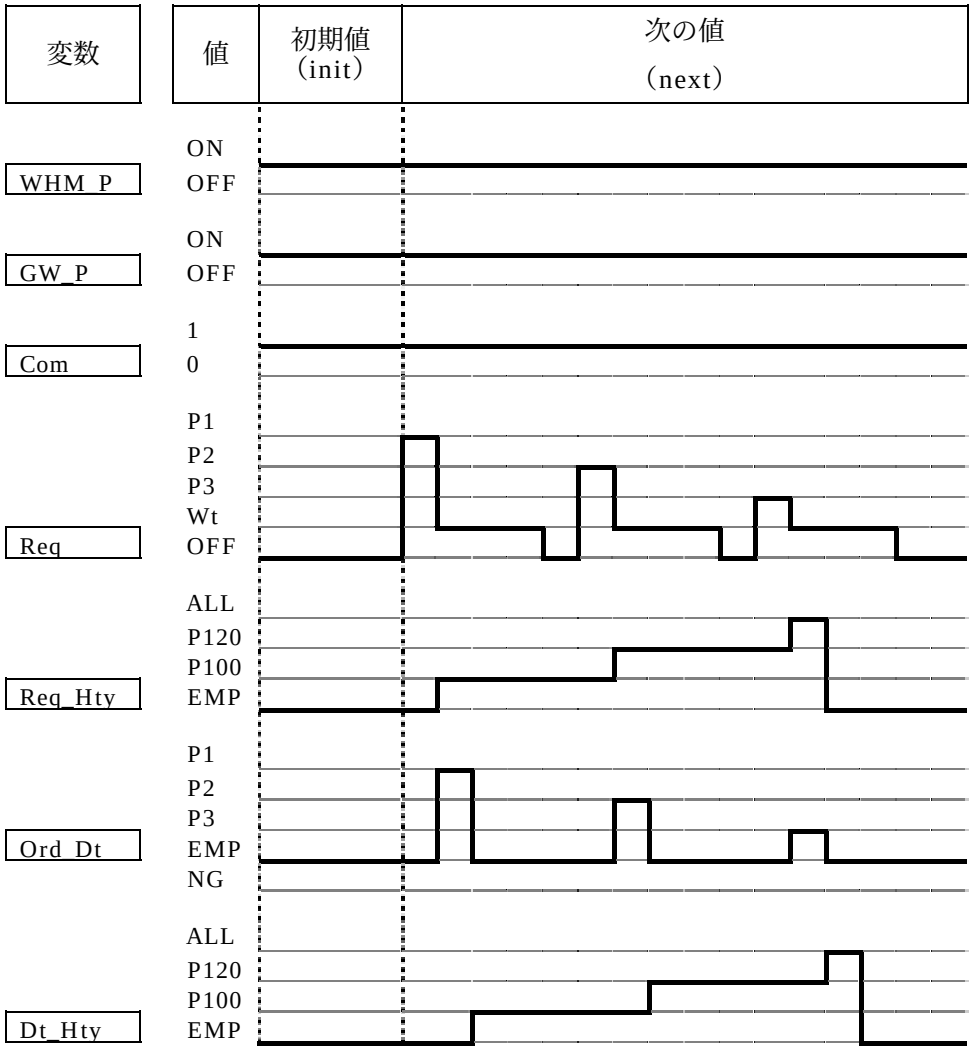


図 22　センタからの要求履歴及びＧＷからの指令データ履歴のタイムチャート

６．コード化

作成した状態遷移図に基づいて仕様のコード化を行った。以下にそのコードを示す。

```
--*****************************************************************************--
--**                                                                         **--
--**                  < The Automatic Meter Reading System SIM >             **--
--**                                                                         **--
--**                          2002/10/22   Kouji Hayamizu                    **--
--**                                                                         **--
--*****************************************************************************--

--*********************************--
--**        Module <Main>        **--
--*********************************--

-- This module is main module for the Automatic Meter Reading System SIM --

MODULE main

VAR
 WHM_P     : {ON, OFF};                        -- Power of WHM                  --
 GW_P      : {ON, OFF};                        -- Power of GW                   --

 Req       : {OFF, M, P1, P2, P3, Wt};         -- Request from Center           --
 Ord_Dt    : {EMP, M, P1, P2, P3, NG};         -- Order data from GW            --
 Ret       : {EMP, OK, NR};                    -- Return from WHM               --

 Rpt       : {EMP, OK, NG};                    -- Report from GW to Center      --

 P_Cnt     : {EMP, P100, P120, P103, ALL};     -- Patern store count in WHM     --
 P_Set     : {ON, OFF};                        -- Patern set flag in WHM        --

 Req_Hty   : {EMP, P100, P120, P103, ALL};     -- Request(P1,P2,P3) history     --
 Dt_Hty    : {EMP, P100, P120, P103, ALL};     -- Ord_Dt(P1,P2,P3) history      --
 Wt_Cnt    : {0,1,2,3,4,5,6,7,8};              -- Wait reset counter for Center --
```

```
DEFINE
 Com       := GW_P = ON  & WHM_P = ON;            -- Communication between WHM & GW   --
 Ord_Dt_P  := Ord_Dt = P1 | Ord_Dt = P2 | Ord_Dt = P3;  -- Patern Ord_Dt for FAIRNESS  --

ASSIGN

--< Wait reset counter for Center >--

 init(Wt_Cnt) := 0;
 next(Wt_Cnt) :=
   case
    Wt_Cnt < 8 & Req = Wt : Wt_Cnt + 1;
    1 : 0;
   esac;

--< Request(P1,P2,P3) history >--

 init(Req_Hty) := EMP;
 next(Req_Hty) :=
   case
    (Req_Hty = EMP | Req_Hty = P120 | Req_Hty = P103 | Req_Hty = ALL) & Req = P1 : P100;
    Req_Hty = P100 & Req = P2 : P120;
    Req_Hty = P100 & Req = P3 : P103;
    Req_Hty = P120 & Req = P3 : ALL;
    Req_Hty = P103 & Req = P2 : ALL;
    Req_Hty = ALL : EMP;
    1 : Req_Hty;
   esac;
```

```
--< Ord_Dt(P1,P2,P3) history >--

 init(Dt_Hty) := EMP;
 next(Dt_Hty) :=
   case
    (Dt_Hty = EMP | Dt_Hty = P120 | Dt_Hty = P103 | Dt_Hty = ALL) & Ord_Dt = P1 : P100;
    Dt_Hty = P100 & Ord_Dt = P2 : P120;
    Dt_Hty = P100 & Ord_Dt = P3 : P103;
    Dt_Hty = P120 & Ord_Dt = P3 : ALL;
    Dt_Hty = P103 & Ord_Dt = P2 : ALL;
    Dt_Hty = ALL : EMP;
    1 : Dt_Hty;
   esac;

--< Power of WHM >--

 init(WHM_P) := ON;
 next(WHM_P) := {ON, OFF};
 --C next(WHM_P) := ON;

 --< Power of GW >--

 init(GW_P) := ON;
 next(GW_P) := {ON, OFF};
 --C next(GW_P) := ON;
```

```
--< Request from Center >--

init(Req) := OFF;
next(Req) :=
  case
   Req = OFF : {M, P1, P2, P3};
   Req = M  : Wt;
   Req = P1 : Wt;
   Req = P2 : Wt;
   Req = P3 : Wt;
   Req = Wt & ( Rpt = OK | Rpt = NG ) : OFF;
   Req = Wt & Wt_Cnt = 7 : OFF;
   1 : Req;
  esac;

--< Order data from GW >--

init(Ord_Dt) := EMP;
next(Ord_Dt) :=
  case
   Ord_Dt = EMP & GW_P = ON & Req = M  : {M , NG};
   Ord_Dt = EMP & GW_P = ON & Req = P1 : {P1, NG};
   Ord_Dt = EMP & GW_P = ON & Req = P2 : {P2, NG};
   Ord_Dt = EMP & GW_P = ON & Req = P3 : {P3, NG};
   --C Ord_Dt = EMP & GW_P = ON & Req = M  : M;
   --C Ord_Dt = EMP & GW_P = ON & Req = P1 : P1;
   --C Ord_Dt = EMP & GW_P = ON & Req = P2 : P2;
   --C Ord_Dt = EMP & GW_P = ON & Req = P3 : P3;
   1 : EMP;
  esac;
```

```
--< Return from WHM >--

 init(Ret) := EMP;
 next(Ret) :=
   case
    Ret = EMP & Com = 1 & ( Ord_Dt = M | Ord_Dt = P1 | Ord_Dt = P2 | Ord_Dt = P3 ) : OK;
    Ret = EMP & Com = 1 & Ord_Dt = NG : NR;
    1 : EMP;
   esac;

 --< Report from GW to Center >--

 init(Rpt) := EMP;
 next(Rpt) :=
   case
    Rpt = EMP & Com = 1 & Ret = OK : OK;
    Rpt = EMP & GW_P = ON & Com = 0 & !(Ord_Dt = EMP) : NG;
    Rpt = EMP & GW_P = ON & Com = 0 & Ret = OK : NG;
    Rpt = EMP & Com = 1 & Ret = NR : NG;
    1 : EMP;
   esac;
```

```
--< Patern store count in WHM >--

 init(P_Cnt) := EMP;
 next(P_Cnt) :=
   case
    (P_Cnt = EMP | P_Cnt = P120 | P_Cnt = P103 | P_Cnt = ALL) & Com = 1 & Ord_Dt = P1 : P100;
    P_Cnt = P100 & Com = 1 & Ord_Dt = P2 : P120;
    P_Cnt = P100 & Com = 1 & Ord_Dt = P3 : P103;
    P_Cnt = P120 & Com = 1 & Ord_Dt = P3 : ALL;
    P_Cnt = P103 & Com = 1 & Ord_Dt = P2 : ALL;
    P_Cnt = ALL & WHM_P = ON & !( Com = 1 & Ord_Dt = P1 ) : EMP;
    1 : P_Cnt;
   esac;

--< Patern set flag in WHM >--

 init(P_Set) := OFF;
 next(P_Set) :=
   case
    P_Set = OFF & WHM_P = ON & P_Cnt = ALL : ON;
    1 : OFF;
   esac;
```