(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2014-206830
(P2014-206830A)
(43)公開日　平成26年10月30日(2014.10.30)

(51)Int.Cl.		F I		テーマコード（参考）
G05B 19/05	(2006.01)	G05B 19/05	A	5H220

審査請求 未請求　請求項の数 9　ＯＬ　（全 21 頁）

(21)出願番号　特願2013-83323(P2013-83323)
(22)出願日　平成25年4月11日(2013.4.11)

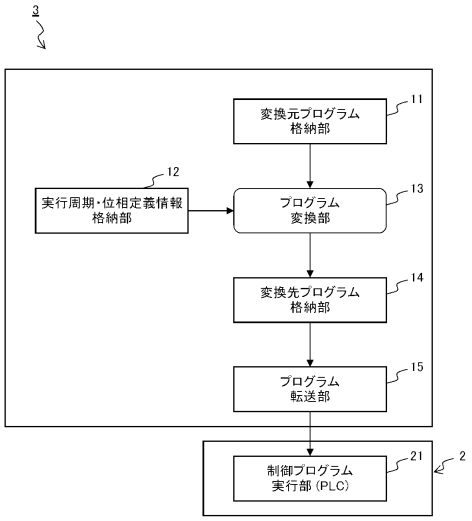
(71)出願人　000005234
富士電機株式会社
神奈川県川崎市川崎区田辺新田１番１号
(74)代理人　100074099
弁理士　大菅　義之
(72)発明者　丸山　吉晴
神奈川県川崎市川崎区田辺新田１番１号
富士電機株式会社内

Ｆターム(参考)　5H220 BB07　CC06　CX02　HH08　JJ12
JJ24　JJ26　JJ34

(54)【発明の名称】プログラマブルコントローラの支援装置、そのプログラム、プログラマブルコントローラシステム

(57)【要約】
【課題】位相制御機能を持たないＰＬＣであっても定周期プログラムの定周期性を守ることが出来る。
【解決手段】“実行周期・位相定義情報格納部”１２には、変換元プログラム格納部１１に格納される複数の“変換元となる定周期プログラム”に係わる実行周期と位相の情報が格納されている。これに基づいて、プログラム変換部１３は、“変換元となる定周期プログラム”から、上記実行周期と位相の情報を反映させた「変換後の定周期プログラム」を生成して、これを位相制御機能を持たないＰＬＣ２にダウンロードさせる。
【選択図】図２

本例の支援装置の機能ブロック図



【特許請求の範囲】
【請求項１】
複数の定周期プログラムを記憶する定周期プログラム記憶手段と、
予め任意に設定された前記各定周期プログラム毎の実行周期と位相を記憶する実行周期・位相記憶手段と、
前記複数の定周期プログラムの前記実行周期及び位相の最大公約数を算出し、前記各定周期プログラム毎に、その前記実行周期と位相と、前記最大公約数とに基づいて、“変換後の定周期プログラム”を生成する“変換後の定周期プログラム”生成手段と、
該各“変換後の定周期プログラム”を、位相制御機能を持たないプログラマブルコントローラに転送して記憶させる転送手段と、
前記プログラマブルコントローラにおける前記各“変換後の定周期プログラム”の実行周期を、前記最大公約数の時間とする実行周期設定手段と、
を有することを特徴とするプログラマブルコントローラの支援装置。
【請求項２】
前記“変換後の定周期プログラム”は、前記定周期プログラムに対して、その前記実行周期と位相と前記最大公約数とに基づいて得られる定義値を用いた所定の条件を満たす場合に該定周期プログラムが実行される処理が、追加されることで生成されるものであることを特徴とする請求項１記載のプログラマブルコントローラの支援装置。
【請求項３】
前記実行周期と位相と最大公約数とに基づいて得られる定義値は、前記実行周期を前記最大公約数で除算することで得られる周期定義値と、前記位相を前記最大公約数で除算することで得られる位相定義値とであることを特徴とする請求項２記載のプログラマブルコントローラの支援装置。
【請求項４】
前記“変換後の定周期プログラム”には、該“変換後の定周期プログラム”の実行回数をカウントする処理が含まれると共に、該実行回数カウント値が前記周期定義値に達する毎に該実行回数カウント値を初期化する処理が含まれることを特徴とする請求項３記載のプログラマブルコントローラの支援装置。
【請求項５】
前記“変換後の定周期プログラム”には、前記実行回数カウント値が初期化された後に該実行回数カウント値が前記位相定義値となったときに、前記定周期プログラムを実行させる処理が含まれることを特徴とする請求項４記載のプログラマブルコントローラの支援装置。
【請求項６】
前記定周期プログラム記憶手段に記憶される複数の定周期プログラムは、位相制御機能を持つプログラマブルコントローラ用に作成されていた定周期プログラムであることを特徴とする請求項１～５の何れかに記載のプログラマブルコントローラの支援装置。
【請求項７】
プログラマブルコントローラの支援装置のコンピュータを、
全ての定周期プログラムの実行周期及び位相の最大公約数を算出し、前記各定周期プログラム毎に、その前記実行周期と位相と、前記最大公約数とに基づいて、“変換後の定周期プログラム”を生成する“変換後の定周期プログラム”生成手段と、
該各“変換後の定周期プログラム”を、位相制御機能を持たないプログラマブルコントローラに転送して記憶させる転送手段と、
前記プログラマブルコントローラにおける前記各“変換後の定周期プログラム”の実行周期を、前記最大公約数の時間とする実行周期設定手段、
として機能させる為のプログラム。
【請求項８】
位相制御機能を持たないプログラマブルコントローラと、プログラマブルコントローラの支援装置とを有するプログラマブルコントローラシステムであって、

前記プログラマブルコントローラの支援装置は、
複数の定周期プログラムを記憶する定周期プログラム記憶手段と、
予め任意に設定された前記各定周期プログラム毎の実行周期と位相を記憶する実行周期・位相記憶手段と、
全ての前記定周期プログラムの前記実行周期及び位相の最大公約数を算出し、前記各定周期プログラム毎に、その前記実行周期と位相と、前記最大公約数とに基づいて、“変換後の定周期プログラム”を生成する“変換後の定周期プログラム”生成手段と、
該各“変換後の定周期プログラム”を、前記位相制御機能を持たないプログラマブルコントローラに転送して記憶させる転送手段と、
前記プログラマブルコントローラにおける前記各“変換後の定周期プログラム”の実行周期を、前記最大公約数の時間とする実行周期設定手段と、
を有することを特徴とするプログラマブルコントローラシステム。
【請求項９】
前記位相制御機能を持たないプログラマブルコントローラは、
前記記憶された全ての前記“変換後の定周期プログラム”を、前記最大公約数の時間とされた前記実行周期で実行する制御プログラム実行手段を有し、
該実行される各“変換後の定周期プログラム”のなかでそのときに所定の条件を満たす“変換後の定周期プログラム”に含まれる定周期プログラムが実行されることを特徴とする請求項８記載のプログラマブルコントローラシステム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、プログラマブルコントローラ（PLC：Programmable Logic Controller）のプログラム実行制御に関する。
【背景技術】
【０００２】
プログラマブルコントローラ（PLC：Programmable Logic Controller）において実行されるプログラムには、アプリケーションプログラムを繰り返し実行するサイクリックプログラムと、予め決められた周期でアプリケーションプログラムを実行する定周期プログラムがある。サイクリックプログラムは特に周期性を必要としない処理に使用され、定周期プログラムは微分や積分などの一定周期で実行しなければならない演算を含む処理で使用される。特に計測制御（プロセス制御）で使用されるＰＩＤ演算では微分/積分処理が含まれる為、必ず定周期プログラムで実行される。
【０００３】
計測制御においては制御対象のプロセスの反応が遅いため、頻繁に演算を行う必要はなく、定周期ではあるが毎周期（タイマ割込周期毎など）の実行ではなく、間欠的に実行すればよい場合がある。つまり、定周期プログラムにおいて、その実行サイクルを、毎周期（例えば定周期タイマ割込処理が発生する毎；タイマ周期毎とも言える；タイマ周期は例えば０．１ｓ毎）ではなく、タイマ周期の２回に１回実行や、タイマ周期の５回に１回実行などと実行回数を間引くことで、１台のコントローラで多くの制御を行うことを可能にしている場合がある。
【０００４】
プログラムを間欠的に実行する方式として、例えば特許文献１に開示されている従来技術が知られている。
但し、上記のように定周期プログラムを間欠的に実行する場合であっても、複数の定周期プログラムの実行サイクルが同一である場合等、実行タイミングが重なると実行時に演算負荷が増大することにより定周期性を守れなくなるなどの問題が発生することがある。
【０００５】
例えば、複数の定周期プログラムの実行タイミングが重なった場合に、図１０（ａ）に

示すように予め決められた順番で順次実行することで（図示に例ではプログラムＡとプログラムＢの実行タイミングが重なった場合、常に、まずＡを実行して続いてＢを実行する）、定周期性が守られる手法がある（図示の例では、プログラムＢは、常に、実行サイクルからプログラムＡの処理時間分遅れたタイミングで実行されるので、実質的に定周期性が守られることになる）。しかしながら、この手法では、プログラムＡの実行時間が変動すると、その直後に実行するプログラムＢの実行タイミングがずれるので、プログラムＢに関しては定周期性を守ることが出来なくなる。

【０００６】

但し、実行タイミングが重なる定周期プログラムの数が、図示のように２つの場合には、この問題はそれほど大きな問題とはならないかもしれない。しかしながら、実行タイミングが重なる定周期プログラムの数が３つや４つとなった場合、最後に実行される定周期プログラムの実行タイミングが、大きくズレるという問題が生じる可能性がある。

【０００７】

この問題に対処するため、例えば図１０（ｂ）に示すように、定周期プログラム毎に異なる位相を設定しておくことで、実行サイクルが同じであっても実行タイミングがズレるようにする手法が知られている、これによって、図示の例の場合、たとえプログラムＡの実行時間が変動した場合でも、プログラムＢの定周期性を守ることができる。

【０００８】

その為に、従来では、例えば図１１（ａ）に示すように、予め各定周期プログラム毎に周期と位相を設定しておくことで（実行周期が同じ定周期プログラム同士は、位相が同じにならないように設定する）、図１１（ｂ）に示すように、プログラムＡとプログラムＢとは、実行周期が同じであっても実行タイミングがズレる。

【０００９】

従来では、例えば図１１（ａ）に示すような周期と位相が設定されていれば、ＰＬＣのＯＳの処理機能によって、各定周期プログラムの実行タイミングを図１１（ｂ）に示すようにすることができる。

【００１０】

この様なＰＬＣのＯＳの処理例を、図１２に示す。尚、これは、ＰＬＣのＯＳが有する“定周期プログラムの間欠実行機能”の処理フローチャートということもできる。

図１２の処理は、定周期（ここでは仮に１００msとする）のタイマ割込みによって起動される。つまり、図１２の処理は、タイマ割込み周期毎（本例では１００（ms）毎）に実行される。しかし、図１２の処理によって、上記のように実質的にタイマ割込み周期の整数倍の周期で、すなわちタイマ割込みの２回に１回（つまり、実質的に２００（ms）周期で）や５回に１回（つまり、実質的に５００（ms）周期で）の割合で、プログラム実行されることになる。

【００１１】

また、図１２の処理実行の為に、ＰＬＣの不図示のメモリには、図１３に示すタスクテーブル１００が格納されている。

図１３に示す例のタスクテーブル１００は、タスク１０１、実行周期１０２、位相１０３、時間１０４の各データ項目より成る。

【００１２】

タスク１０１は、各プログラム（タスク）の名称や識別用ＩＤ等である。ここでは、仮に、プログラムＡ，Ｂ，Ｃの３つのプログラム（タスク）が登録されているものとする。

実行周期１０２と位相１０３は、各プログラム（Ａ，Ｂ，Ｃ）の実行周期と位相である。尚、ここでは、実行周期毎の任意のタイミング（例えば８００ms毎のタイミング）を、実行周期タイミングと呼ぶものとする。任意の実行周期タイミングから例えば８００（ｍｓ）経過した時点が、次の実行周期タイミングとなる。基本的には（プログラムＣの場合には）実行周期タイミングがプログラム（タスク）実行のタイミングとなるが、本例ではプログラムＡ，Ｂについては位相によって実行タイミングをズラしている。

【００１３】

つまり、各プログラムは、実行周期１０２毎（例えば８００ms毎）に実行されるが、実行周期タイミングから位相１０３経過したタイミングで実行される。尚、タスク１０１と実行周期１０２と位相１０３に係わる図示のデータ例は、図１１に示す例に対応している。つまり、例えばプログラムＢは、図１１（ｂ）に示すタイミングで実行されることになる。

【００１４】

尚、各実行周期１０２は、基本的に、タイマ割込み周期の整数倍となるように設定される。

時間１０４には、そのプログラムの直近の実行周期タイミングからの経過時間が格納される。よって、時間１０４は、随時更新されると共に、そのプログラムの実行周期タイミングとなる毎に‘０’リセットされる。

【００１５】

尚、本説明における「実行周期」は、所謂「制御周期」（ＰＩＤ等の制御演算周期）と同義と見做しても構わないが、この例に限らない。

図１２に処理では、まず、タスクテーブル１００の全てのプログラム（全てのレコード）の時間１０４を更新する（例えば、タイマ割込み周期分を加算する；本例では＋１００（ms）インクリメントする）（ステップＳ５１）。

【００１６】

そして、タスクテーブル１００の各プログラム（各レコード）を順次処理対象として、処理対象プログラムについてステップＳ５２～Ｓ５６の処理を実行することを繰り返す（ループ処理）。そして、全てのプログラムについて処理実行したら（ステップＳ５６，ＹＥＳ）本処理を終了する。

【００１７】

上記ループ処理では、まず、処理対象プログラムの実行周期１０２と時間１０４とを比較して、両者が一致するか否かを判定する（ステップＳ５２）。両者が一致する場合には（ステップＳ５２，ＹＥＳ）、それは現時点が処理対象プログラムの実行周期タイミングであることを意味し、時間１０４を‘０’リセットする（ステップＳ５３）。

【００１８】

続いて、上記ステップＳ５２の判定結果に係わらず、処理対象プログラムの位相１０３と時間１０４とを比較して、両者が一致するか否かを判定する（ステップＳ５４）。両者が一致する場合には（ステップＳ５４，ＹＥＳ）、処理対象プログラムの実行タイミングであると見做して、当該プログラムを実行させる（ステップＳ５５）。尚、本例では、任意のプログラムを実行させる際にはこのプログラムの識別情報等を所定のキューに格納するものとする。勿論、ＰＬＣのＯＳには、所定のキューからデータを取り出してプログラム実行する機能が、別途、備わっているが、これについては特に説明しない。

【００１９】

尚、図１３には、タスクテーブル１００の図上右側に、時間１０４の更新例とプログラム実行タイミングを示している。尚、星印（☆）で示す時点が、各プログラムの実行タイミングであり、上記ステップＳ５５の処理が行われていることになる。例えば、プログラムＣを例にすると、実行周期１０２が２００（ms）であることから、時間１０４が２００（ms）となる毎に時間１０４が‘０’にリセットされると共に、当該時間１０４（＝０）と位相（＝０ms）とが一致することから、ステップＳ５５の処理が実行されることになる。他のプログラムについても略同様である。

【００２０】

また、上記の通り、図１３に示すデータ例に基づいて図１２の処理を実行すると、各プログラムの実行タイミングは図１１（ｂ）に示すようになるが、図示のように２つのプログラム（プログラムＣと他のプログラム）の実行タイミングが重なる場合があるが、同時に実行されるわけではない。すなわち、各プログラムＡ，Ｂ，Ｃは、例えば図１４に示すタイミングで実行される。

【００２１】

尚、ここでは、予め各プログラムに優先順位（タスクレベル）が割り当てられており、複数のプログラムの実行タイミングが重なった場合、タスクレベルが高いものから順に実行されるものとする。尚、図１４に示す例では、タスクレベルは「Ａ＞Ｂ＞Ｃ」であるものとする。

【００２２】

よって、図１４に示すように、例えばプログラムＡとプログラムＣの実行タイミングが重なった場合、まずプログラムＡが実行されて処理終了したら直ちにプログラムＣが実行される。この場合、上記のように、プログラムＡの実行時間が変動するとプログラムＣの実行開始が変動するので、プログラムＣの定周期性に多少影響するが、それだけでは問題とはならない。しかし、プログラムＡ，Ｂ，Ｃの実行タイミングが重なった場合、プログラムＡ，Ｂの実行時間が変動することで、プログラムＣの定周期性に大きく影響する可能性がある。それ故、従来では上記のように位相を用いることで、実行タイミングをズラして、実行タイミングが重なるプログラム数を減少させている。

【先行技術文献】

【特許文献】

【００２３】

【特許文献１】特開平０７－２０００１２号公報

【発明の概要】

【発明が解決しようとする課題】

【００２４】

ここで、従来より、例えば計測制御で用いられるプログラマブルコントローラでは、そのＯＳに上記図１２の処理機能が実装されている。計測制御では、上記実行周期と位相により、制御処理の時間的な均一化を図っている。

【００２５】

一方、機械制御などのシーケンス制御では、特に定周期性は必要なく、入力に応じて逐次演算を実行して出力を行っていけばよく、その繰り返しを高速に行うことが求められており、従来はサイクリックプログラム実行が中心であった。

【００２６】

しかし、機械制御でも定期的な実行が求められる処理があり、それは定周期プログラムで実行されている。しかし、機械制御分野では制御対象の応答が高速であるため、プログラムの間欠実行は求められていないことが多く、ＰＬＣにこのような間欠実行機能が実装されていないことがある。例えば、ＰＬＣのプログラム言語の国際標準であるIEC61131-3においては、このような間欠実行機能は定義されていない（定周期プログラムは定義されている）。つまり、IEC61131-3標準に対応したＰＬＣの場合、上記間欠実行機能は実装されていない。

【００２７】

尚、ここでいう“間欠実行機能”とは、例えば図１２の処理で実現させるような機能であり、周期設定だけでなく、例えば位相設定等によって実行タイミングをズラす機能である。尚、その意味で、“間欠実行機能”は、“位相制御機能”と言ってもよいし、あるいは“位相制御機能”を含むものと見做してもよい。

【００２８】

従って、周期設定だけによって単にタイマ割込周期の数倍（２回に１回など）をプログラム実行周期とする機能だけでは、そのＰＬＣは“間欠実行機能”（“位相制御機能”）は有していないものと見做すものとする。但し、上記位相を用いるのは一例であり、この例に限らない。すなわち、“間欠実行機能”とは、そのプログラムの実行タイミングを実行周期タイミングからズラすようにして決定する機能であると言える。

【００２９】

この為、定周期プログラムの間欠実行機能により制御を行っていたＰＬＣのプログラムを、間欠実行機能を持たないＰＬＣ（上記IEC61131-3標準に対応したＰＬＣ等）で実行させたい要望があっても、この要望は容易には実現できなかった（少なくとも単にプログラ

ムを移植するだけでは実現できなかった）。
【００３０】
尚、IEC61131-3標準に対応したＰＬＣでは間欠実行機能を実現できない理由は、そのＯＳの機能が位相に対応していない為である（周期には対応している）。
本発明の課題は、位相制御機能を持たないＰＬＣ（例えばIEC61131-3標準に対応したＰＬＣ）であっても定周期プログラムの定周期性を守ることが出来るプログラマブルコントローラの支援装置等を提供することである。
【課題を解決するための手段】
【００３１】
本発明のプログラマブルコントローラの支援装置は、複数の定周期プログラムを記憶する定周期プログラム記憶手段と、予め任意に設定された前記各定周期プログラム毎の実行周期と位相を記憶する実行周期・位相記憶手段と、全ての前記定周期プログラムの前記実行周期及び位相の最大公約数を算出し、前記各定周期プログラム毎に、その前記実行周期と位相と、前記最大公約数とに基づいて、“変換後の定周期プログラム”を生成する“変換後の定周期プログラム”生成手段と、該各“変換後の定周期プログラム”を、位相制御機能を持たないＰＬＣに転送して記憶させる転送手段と、前記ＰＬＣにおける前記各“変換後の定周期プログラム”の実行周期を、前記最大公約数の時間とする実行周期設定手段とを有する。
【発明の効果】
【００３２】
本発明のプログラマブルコントローラの支援装置等によれば、位相制御機能を持たないＰＬＣ（例えばIEC61131-3標準に対応したＰＬＣ）であっても定周期プログラムの定周期性を守ることが出来る。
【図面の簡単な説明】
【００３３】
【図１】プログラマブルコントローラシステムの概略構成図である。
【図２】本例の支援装置の機能ブロック図である。
【図３】「変換後の定周期プログラム」の雛形（フォーマット）の例である。
【図４】プログラム変換部の処理フローチャート図である。
【図５】（ａ）、（ｂ）は、位相・周期定義表の具体例（その１）、（その２）である。
【図６】（ｂ）、（ｃ）は周期定義値、位相定義値の具体例（その１）、（その２）であり、（ａ）は「変換後の定周期プログラム」の具体例である。
【図７】ＰＬＣのＯＳの処理フローチャート図である。
【図８】「補正後のタスクテーブル」の一例である。
【図９】（ａ）、（ｂ）は、処理実行の有無等に係わる具体例である。
【図１０】（ａ）、（ｂ）は、実行周期が同じプログラムの実行処理例である。
【図１１】（ａ），（ｂ）は、従来において位相を用いて実行タイミングをズラす例である。
【図１２】従来のＰＬＣのＯＳの処理フローチャート図である。
【図１３】従来のＰＬＣ側で保持するタスクテーブルの一例である。
【図１４】従来のプログラム実行タイミング例である。
【発明を実施するための形態】
【００３４】
以下、図面を参照して、本発明の実施の形態について説明する。
図１は、本例のプログラマブルコントローラ（ＰＬＣ）やその支援装置等から成るプログラマブルコントローラシステムの概略システム構成図である。
【００３５】
図示のＰＬＣシステム１は、ＰＬＣ２と支援装置３とが通信線４等によって接続された構成である。
支援装置３は、ＰＬＣ用のプログラム（特に上記定周期プログラム等）を開発者に任意

に作成させるように支援するプログラム作成支援機能を有する。また、ＰＬＣ用のプログラムをＰＬＣ２にダウンロードして記憶させるダウンロード機能を有する。尚、これらプログラム作成支援機能やダウンロード機能自体は、従来の支援装置と略同様であってよく、ここでは特に説明しない。

【００３６】

ここで、ＰＬＣ２は、上記“間欠実行機能（位相制御機能）を持たないＰＬＣ”（例えば上記IEC61131-3標準に対応したＰＬＣ等）である。また、後述する“変換元となる定周期プログラム”は、例えば間欠実行機能を有するＰＬＣ用に作成されていたプログラムである。従って、“変換元となる定周期プログラム”をそのままＰＬＣ２にダウンロードして実行させても、位相をズラしつつ間欠実行することは行われないことになる（正常動作しないと見做してよい）。

【００３７】

尚、上述したように、ここでは、“間欠実行機能”とは、例えば上記図１２の処理で実現させるような機能であり、周期設定だけでなく、例えば位相設定等によって実行タイミングをズラす機能である。尚、その意味で、“間欠実行機能”は、“位相制御機能”と言ってもよいし、あるいは“位相制御機能”を含むものと見做してもよい。

【００３８】

尚、本説明において、「変換」を「補完」に置き換えても構わない。後述するように、本手法では、“変換元となる定周期プログラム”に対して後述するステップＳ１，Ｓ２，Ｓ３，Ｓ５の処理を追加したり後述する実行周期５２を補正することで（つまり、ＰＬＣ２で正常操作させる為に必要な処理やデータを補完することで）、後述する「変換後の定周期プログラム」を生成するものであるからである。

【００３９】

何れにしても、本手法における“定周期プログラムの「変換」（または「補完」）”とは、一例として後述する図６（ａ）のような「変換後の定周期プログラム」を生成することである（更に、実行周期５２を補正することである）。つまり、例えば位相制御機能を持つＰＬＣ用に作成されていた複数の定周期プログラムを、位相制御機能を持たないＰＬＣ２（例えばIEC61131-3標準に対応したＰＬＣ）で問題なく実行できる（定周期性を守ることが出来る）ようにする為に、“定周期プログラムの「変換」（または「補完」）”を行うものである。

【００４０】

尚、既に述べたように、“間欠実行機能”は、そのプログラムの実行タイミングを実行周期タイミングからズラすようにして決定する（実行タイミングが出来るだけ他のプログラムと重複しないようにする）機能（位相制御機能）も有する。従って、実行周期だけによって単にタイマ割込周期の整数倍の周期で実行させるだけでは、“間欠実行機能”（位相制御機能）は有していないものと見做すものとする。

【００４１】

本例の支援装置３は、更に、定周期プログラムの変換（補完）機能を有する。

これについて、図２を参照して説明する。

図２は、主に支援装置３の機能ブロック図である（但し、ＰＬＣ２の機能も一部示されている）。

【００４２】

図示の例では、支援装置３は、変換元プログラム格納部１１、“実行周期・位相定義情報格納部”１２、プログラム変換部１３、変換先プログラム格納部１４、プログラム転送部１５等を有する。

【００４３】

変換元プログラム格納部１１には、変換元となる任意の複数の定周期プログラムが格納される。これら“変換（補完）元となる定周期プログラム”は、例えば間欠実行機能を有するＰＬＣ用に作成されていたプログラムである。つまり、一例としては、上記従来技術の説明で用いた定周期プログラムＡ，Ｂ，Ｃであってよく、以下、具体例としてはこの例

を用いて説明するものとする。

【００４４】

尚、上記複数の“変換元となる定周期プログラム”は、当該支援装置３において作成されたものであってもよいし、他の何らかの情報処理装置で作成されたもの（間欠実行機能を有するＰＬＣで運用されていたものであっても構わない）が、何等の方法で当該支援装置３に格納されたものであってもよい。

【００４５】

ここで、ダウンロード先のＰＬＣが上記間欠実行機能を有する不図示のＰＬＣである場合には、変換元プログラム格納部１１に格納されている上記複数の“変換元となる定周期プログラム”を、そのまま、当該ＰＬＣにダウンロードすればよい（ＰＬＣのＯＳの機能によって、図１１（ｂ）等に示す例のように定周期プログラムが実行される）。尚、逐一述べないが、当該ＰＬＣは、当然、上記タスクテーブル１００に相当する情報を保持している。

【００４６】

一方、ダウンロード先が上記ＰＬＣ２のような“間欠実行機能を持たないＰＬＣ”である場合には、上記複数の“変換元となる定周期プログラム”（Ａ，Ｂ，Ｃ）を、そのまま、ＰＬＣ２にダウンロードした場合、ＰＬＣ２のＯＳは位相に応じて実行タイミングをズラすことは出来ないので、図１１（ｂ）等に示す例のような実行タイミングを実現できない。これは、ＰＬＣ２が上記タスクテーブル１００に相当する情報（各艇周期プログラム毎の周期と位相の設定情報等）を保持していても実現できない。位相の設定情報があっても、ＯＳにはそれを活かす機能が無いからである。

【００４７】

尚、ＰＬＣ２のＯＳの処理例は、図７に示し、後に説明する。

この問題に対して、本手法では、主に“実行周期・位相定義情報格納部”１２とプログラム変換部１３を設けている。

【００４８】

“実行周期・位相定義情報格納部”１２には、上記複数の“変換元となる定周期プログラム”に係わる実行周期と位相の情報（後述する位相・周期定義表等）が格納されている。尚、この情報は上記タスクテーブル１００に相当すると見做してもよい（但し、上記時間１０４は必要ない）。

【００４９】

プログラム変換部１３は、上記“実行周期・位相定義情報格納部”１２の格納データ等に基づいて、上記複数の“変換元となる定周期プログラム”をそれぞれ変換して、これら各「変換（補完）後の定周期プログラム」を変換先プログラム格納装置１４に格納する。この変換方法については後述する。

【００５０】

プログラム転送部１４は、上記変換先プログラム格納装置１４に格納されたプログラム（変換後の定周期プログラム）を、上記通信線４等を介してＰＬＣ２にダウンロードして記憶させる。

【００５１】

また、プログラム変換部１３は、上記変換処理に伴って、ＰＬＣ２側で保持させるタスクテーブル等の内容を補正する。これは特に実行周期を補正するものであり、詳しくは後述する。尚、ＰＬＣ２側で保持されているタスクテーブルは、上記従来のタスクテーブル１００と略同様であっても構わない（但し、位相の情報は無くても構わない。また補正を行う必要がある）。

【００５２】

尚、この補正処理は、例えば上記タスクテーブルを既にＰＬＣ２側で保持している場合には、その実行周期に対して上記変換処理に応じた補正を行うことで、例えば後述するタスクテーブル５０を生成する。一方、タスクテーブルがＰＬＣ２側に保持されていない場合には、例えば上記“実行周期・位相定義情報格納部”１２の格納データにおける実行周

期に対して上記変換処理に応じた補正を行うこと等によって後述するタスクテーブル５０を生成し、これをＰＬＣ２にダウンロードして記憶させる。

【００５３】

尚、後述するタスクテーブル５０は、「補正後のタスクテーブル」の一例ということもできる。

何れにしても、ＰＬＣ２には、上記「変換後の定周期プログラム」と「補正後のタスクテーブル」等が記憶された状態になる。そして、ＰＬＣ２が有する制御プログラム実行部２１は、これら「変換後の定周期プログラム」と「補正後のタスクテーブル」を用いて定周期プログラム実行制御を行うことで、実質的に、定周期プログラムの間欠実行を実現できる。

【００５４】

尚、プログラム変換部１３の変換処理は、主に例えば後述する図３のフォーマットへと変換するものであるが、これだけに限らず、変換元／変換先のプログラム言語の組み合わせに応じた適切な変換処理が行われるものであっても構わない（但し、これについては特に説明しない）。

【００５５】

尚、変換元プログラム格納部１１と変換先プログラム格納部１４には、そのプログラムを実行するプログラマブルコントローラ固有の方式によってプログラムが格納されている。

【００５６】

尚、本説明における「実行周期」は、所謂「制御周期」（ＰＩＤ等の制御演算周期）と同義と見做しても構わないが、この例に限らない。

以下、具体例等を用いながら更に詳しく説明する。

【００５７】

まず、図３に、上記「変換後の定周期プログラム」の雛形（フォーマット）の例を示す。

例えば図３に示すような「変換後の定周期プログラム」の雛形が、予め作成されて支援装置３内の不図示の記憶装置（ハードディスク等）に記憶されている。そして、この雛形などに基づいて、各「変換後の定周期プログラム」が作成される。

【００５８】

図３において、ステップＳ４の「制御処理」が、上記「変換元となる定周期プログラム」の処理に相当する。つまり、基本的には、プログラム変換部１３が、「変換元となる定周期プログラム」に対して、図３のステップＳ１，Ｓ２，Ｓ３，Ｓ５の処理を付加することによって、上記「変換後の定周期プログラム」が生成されることになる。

【００５９】

但し、プログラム変換部１３は、ステップＳ１の周期定義値と、ステップＳ３の位相定義値について、各定周期プログラム毎に実行周期や位相等に応じた値を決定したうえで、ステップＳ１，Ｓ２，Ｓ３，Ｓ５の処理を付加する。例えば、上記定周期プログラムＡに応じた周期定義値と位相定義値の具体値を決定したうえで、定周期プログラムＡの処理をステップＳ４の処理としてステップＳ１，Ｓ２，Ｓ３，Ｓ５の処理を付加することで、「変換後の定周期プログラムＡ」が生成されることになる。他の定周期プログラムＢ，Ｃについても同様にして、「変換後の定周期プログラムＢ」、「変換後の定周期プログラムＣ」が生成されることになる。

【００６０】

ここで、図３に示す例では、上記ステップＳ１の処理は「実行カウンタ＝周期定義値？」の判定処理であり、ステップＳ２は「実行カウンタ０クリア」である。ステップＳ１の判定がＹＥＳの場合にステップＳ２の処理が実行される。ステップＳ３は「実行カウンタ＝位相定義値？」の判定処理であり、ステップＳ３の判定がＹＥＳの場合に上記「制御処理」（ステップＳ４）が実行される。最後に、上記実行カウンタを＋１インクリメントする処理（ステップＳ５）が実行される。

【００６１】
図４は、プログラム変換部１３の処理フローチャートを示すものである。
図４の処理例では、まず、“実行周期・位相定義情報格納部”１２に格納されている位相・周期定義表に基づいて、実行周期及び位相の最大公約数を求める（ステップＳ１１）。これは、例えば、全ての定周期プログラム（Ａ，Ｂ，Ｃ）の実行周期及び位相に基づいて、これらの最大公約数を求めるものである。
【００６２】
そして、各「変換元となる定周期プログラム」を、順次、処理対象プログラムとして、各処理対象プログラム毎に、当該プログラムの実行周期、位相や上記最大公約数等に基づいて、上記周期定義値と位相定義値を決定する。そして、決定した周期定義値と位相定義値とを用いた上記Ｓ１，Ｓ３の処理を生成する（ステップＳ１２）。
【００６３】
ここで、周期定義値、位相定義値の算出式は、例えば下記の通りであるが、この例に限らない。
・周期定義値＝実行周期÷最大公約数
・位相定義値＝位相÷最大公約数
そして、処理対象プログラムの処理を上記ステップＳ４とし、当該ステップＳ４に対して上記ステップＳ１，Ｓ２，Ｓ３，Ｓ５の処理を図３のフォーマットに従って追加することで、処理対象プログラムを上記「変換後の定周期プログラム」に変換する（ステップＳ１３）。
【００６４】
尚、全ての定周期プログラムを処理対象プログラムとして上記変換処理を実施したならば（ステップＳ１４，ＹＥＳ）、ループ処理を抜けてステップＳ１５の処理を実行して本処理は終了する。ステップＳ１５の処理については後述する。
【００６５】
上記ステップＳ１１、Ｓ１２の処理について、図５に示す具体例を用いて説明する。
図５（ａ）、（ｂ）には、上記位相・周期定義表の具体例（その１）、（その２）を示す。
【００６６】
尚、位相・周期定義表３０は、そのデータ構造に関しては、タスク３１、実行周期３２、位相３３等を有するものであり、これらは上記従来のタスク１０１、実行周期１０２、位相１０３と略同様と見做してよい。
【００６７】
そして、図５（ａ）に示す具体例（その１）は、上記従来のタスクテーブル１００のデータ例と略同様のデータ内容となっている。すなわち、図示の例の位相・周期定義表３０では、プログラムＡは実行周期３２が８００（ms）で位相３３が２００（ms）、プログラムＢは実行周期３２が８００（ms）で位相３３が４００（ms）、プログラムＣは実行周期３２が２００（ms）で位相３３が０（ms）となっている。
【００６８】
この例の場合（０は除外するものとする）、ステップＳ１１では８００と４００と２００との最大公約数を求めることになるので、最大公約数＝２００となる。
これより、上記周期定義値、位相定義値の算出式を用いた場合、各定周期プログラムＡ，Ｂ，Ｃそれぞれの周期定義値、位相定義値は、図６（ｂ）に示すようになる。一例として、定周期プログラムＡを用いて説明するならば、その実行周期は８００（ms）、位相は２００（ms）であるので、
・周期定義値＝８００÷２００＝４
・位相定義値＝２００÷２００＝１
となる。
【００６９】
そして、この例の場合、「変換後の定周期プログラムＡ」は、図６（ａ）に示すように

なる。
すなわち、この例では、ステップＳ１は「実行カウンタ＝４？」となり、ステップＳ３は「実行カウンタ＝１？」となる。
【００７０】
尚、特に図示しないが、「変換後の定周期プログラムＢ」におけるステップＳ１は「実行カウンタ＝４？」となり、ステップＳ３は「実行カウンタ＝２？」となる。変換後の定周期プログラムＣにおけるステップＳ１は「実行カウンタ＝１？」となり、ステップＳ３は「実行カウンタ＝０？」となる。
【００７１】
尚、これら各「変換後の定周期プログラム」を用いるＰＬＣ２の動作については、後に説明するものとする。
次に、図５（ｂ）に示す具体例（その２）について説明する。
【００７２】
具体例（その２）は、上記具体例（その１）とほぼ同じであり、違いはプログラムＡの位相が３００（ms）となっている点だけである。
この例の場合には、ステップＳ１１では８００と４００と３００と２００との最大公約数を求めることになるので、最大公約数＝１００となる。
【００７３】
そして、最大公約数＝１００を用いて上記の周期定義値、位相定義値の算出処理等を行うことになり、周期定義値、位相定義値は例えば図６（ｃ）に示す通りとなる。そして、例えば図６（ｃ）に示す周期定義値、位相定義値に基づいて各「変換後の定周期プログラム」が生成されるが、これについては説明を省略する。
【００７４】
ここで、ステップＳ１５の処理について説明する。
ここでは、ＰＬＣ２には既に上記位相・周期定義表３０と略同一のタスクテーブルを既に保持しているものとする。そして、上記具体例（その１）を例にするならば、ＰＬＣ２は上記タスクテーブル１００を保持しているものと見做してもよいものとする。
【００７５】
この例の場合、ステップＳ１５の処理は、上記最大公約数を用いてタスクテーブル１００の実行周期１０２を補正する処理となる。図１３に示す例では、タスクテーブル１００の各実行周期１０２は、８００（ms）、８００（ms）、２００（ms）となっているが、これらを全て最大公約数（＝２００（ms））とする補正を行う。これによって後述するタスクテーブル５０（補正後のタスクテーブル）が生成されることになる。
【００７６】
尚、ステップＳ１５の処理は上記一例に限らない。例えば、上記位相・周期定義表３０のコピーに対して、その実行周期を全て最大公約数（＝２００（ms））とする補正を行うことで、「補正された位相・周期定義表（コピー）」を生成して、これをＰＬＣ２にダウンロードして（時間５４を付加して）上記タスクテーブル５０として記憶させるように構成してもよい。
【００７７】
以下、上記「変換後の定周期プログラム」や「補正後のタスクテーブル」を保持している状態のＰＬＣ２の動作について説明する。
ここで、本例のＰＬＣ２は、上記の通り、“間欠実行機能を持たないＰＬＣ”であるので、そのＯＳの機能・動作は、上記従来の図１２等とは異なることになる。
【００７８】
図７に、ＰＬＣ２のＯＳ（オペレーティングシステム）の処理フローチャート図を示す。
尚、図７の処理は、概略的には、上記図１２の処理においてステップＳ５４の処理が無いものと略同様と見做しても構わない。また、図７の処理は、図１２と同様に、定周期タイマ割込みによって１００（ms）毎に起動・実行されるものとする。

【００７９】
　また、図８には、上記「補正後のタスクテーブル」の一例を示す。
　「補正後のタスクテーブル」５０は、データ構造自体は、上記タスクテーブル１００と略同様と見做してもよい。但し、格納されるデータの内容が、上記の通り補正されている。
【００８０】
　図８に示す例の「補正後のタスクテーブル」５０は、タスク５１、実行周期５２、位相５３、時間５４の各データ項目より成る。これら各データ項目の意味は、上記タスクテーブル１００のタスク１０１、実行周期１０２、位相１０３、時間１０４と略同様と見做してよく、ここでの説明は省略する。但し、全てのレコードの実行周期５２が（上記の通り）最大公約数（＝２００（ms））とする補正が行われている。
【００８１】
　図７の処理例では、まず、「補正後のタスクテーブル」５０の全レコードの時間５４を更新する（例えば、タイマ割込み周期分を加算する；よって本例では＋１００（ms）インクリメントする）（ステップＳ２１）。
【００８２】
　そして、「補正後のタスクテーブル」５０の各レコード（各定周期プログラム）を、順次処理対象として、ステップＳ２２～２４の処理を繰り返し実行する。そして、全レコードについて処理実行したら（ステップＳ２５，ＹＥＳ）本処理を終了する。
【００８３】
　ステップＳ２２～２４の処理について以下に説明する。
　まず、処理対象レコードの実行周期５２と時間５４とが一致するか否かを判定する（ステップＳ２２）。これは、処理対象レコードに係わる上記「変換後の定周期プログラム」の実行タイミングとなったか否かを判定するものと言うこともできる。但し、本手法では、「変換後の定周期プログラム」の実行タイミングとなっても、それがその定周期プログラムの実質的な実行タイミングを意味するとは限らない。つまり、「変換後の定周期プログラム」の実行タイミングとなる毎に「変換後の定周期プログラム」は実行されるが、そのステップＳ４の処理が実行されるとは限らない。
【００８４】
　尚、各「変換後の定周期プログラム」の実行に関して、例えばタスクテーブル５０の各定周期プログラム毎に対応する上記“実行カウンタ”が設けられるものとする（例えばタスクテーブル５０には不図示の実行カウンタ５５のデータ項目が更にあってもよい）。
【００８５】
　実行周期５２と時間５４とが一致する場合には（ステップＳ２２，ＹＥＳ）、処理対象レコードに係わる上記「変換後の定周期プログラム」の識別情報等を、不図示のキュー等に格納することで、当該「変換後の定周期プログラム」を実行させる（ステップＳ２３）。更に、処理対象レコードの時間５４を‘０’にリセットして（ステップＳ２４）、ステップＳ２５に移行する。一方、実行周期５２と時間５４とが不一致の場合には（ステップＳ２２，ＮＯ）、そのままステップＳ２５へ移行する。
【００８６】
　ここで、本手法では、上記ステップＳ２３によって「変換後の定周期プログラム」が実行されても、実質的にその定周期プログラムは実行されない（つまり、上記ステップＳ４の処理が実行されない）場合がある。
【００８７】
　ここで、図９（ａ）には、任意の経過時間（ms）に応じた上記各レコードの時間５４の値とステップＳ２３、Ｓ２４の処理の実行の有無（実行する場合は三角印（△）を記す）を示している。図９（ｂ）には図９（ａ）に示す例に応じた実行カウンタの値と実質的な定周期プログラム実行の有無（ステップＳ４の実行の有無（実行する場合は星印（☆）を記す）を示している。
【００８８】

上記のようにタスクテーブル５０の実行周期５２は全プログラム一律“２００（ms）”（最大公約数）となるように補正されていることから、図９（ａ）に示すように、全ての「変換後の定周期プログラム」は２００（ms）周期で実行されるようになる。但し、図９（ｂ）に示すように、三角印（△）があるタイミングであっても星印（☆）があるとは限らない。

【００８９】

この例では、経過時間＝０（前回の８００（ms）に相当すると見做してもよい）において全プログラムＡ，Ｂ，ＣでステップＳ１の判定がＹＥＳとなって（当然、ステップＳ２３が行われている）各実行カウンタが‘０’にリセットされている（但し、ステップＳ５で‘１’になっている）ものとする。

【００９０】

これより、まず、経過時間＝１００（ｍｓ）のタイミングでは、全レコードでステップＳ２１で時間５４＝１００（ｍｓ）となり、且つ、全レコードで実行周期５２＝２００（ｍｓ）であるので、図示のように全レコードでステップＳ２２の判定はＮＯとなる（よって三角印（△）は無い）。また、これより、全プログラムＡ，Ｂ，Ｃの実行カウンタは‘１’のままである。

【００９１】

次に、経過時間＝２００（ｍｓ）のタイミングでは、全レコードでステップＳ２１で時間５４＝２００（ｍｓ）となり、且つ、全レコードで実行周期５２＝２００（ｍｓ）であるので、図示のように全レコードでステップＳ２２の判定はＹＥＳとなる（よって三角印（△）がある）。

【００９２】

これより、「変換後の定周期プログラムＡ」、「変換後の定周期プログラムＢ」、「変換後の定周期プログラムＣ」が順次実行されることになる。尚、これに伴って、全レコードで時間５４が‘０’にリセットされる（ステップＳ２４）。

【００９３】

ここで、この時点では全ての「変換後の定周期プログラム」に関して、その実行カウンタは‘１’となっている。

これより、まず、「変換後の定周期プログラムＡ」におけるステップＳ３の判定はＹＥＳとなるので（位相定義値＝１なので）、ステップＳ４の処理が実行される（定周期プログラムＡの実質的な実行が成される；よって星印（☆）がある）。そして、ステップＳ５で実行カウンタが＋１インクリメントされる（実行カウンタ＝２となる）。

【００９４】

また、「変換後の定周期プログラムＢ」におけるステップＳ３の判定はＮＯとなるので（位相定義値＝２なので）、ステップＳ４の処理が実行されることなく、ステップＳ５が実行される（実行カウンタ＝２となる）。

【００９５】

「変換後の定周期プログラムＣ」の実行に関しては、まず、ステップＳ１の判定がＹＥＳとなるので（周期定義値＝１なので）、実行カウンタは０クリアされる。これより、ステップＳ３の判定はＹＥＳとなるので（位相定義値＝０なので）、ステップＳ４の処理が実行され（定周期プログラムＣの実質的な実行が成される；よって星印（☆）がある）、ステップＳ５が実行される（実行カウンタ＝１となる）。

【００９６】

次に、経過時間＝３００（ｍｓ）のタイミングでは、全レコードでステップＳ２１で時間５４＝１００（ｍｓ）となり、且つ、全レコードで実行周期５２＝２００（ｍｓ）であるので、図示のように全レコードでステップＳ２２の判定はＮＯとなる（よって三角印（△）は無い）。また、これより、プログラムＡ，Ｂの実行カウンタは‘２’のままであり、プログラムＣの実行カウンタが‘１’のままである。

【００９７】

次に、経過時間＝４００（ｍｓ）のタイミングでは、全レコードでステップＳ２１で時

間５４＝２００（ｍｓ）となり、且つ、全レコードで実行周期５２＝２００（ｍｓ）であるので、図示のように全レコードでステップＳ２２の判定はＹＥＳとなる（よって三角印（△）がある）。

【００９８】

これより、「変換後の定周期プログラムＡ」、「変換後の定周期プログラムＢ」、「変換後の定周期プログラムＣ」が順次実行されることになる。尚、これに伴って、全レコードで時間５４が‘０’にリセットされる（ステップＳ２４）。

【００９９】

ここで、この時点では各「変換後の定周期プログラム」に関して、プログラムＡ，Ｂの実行カウンタは‘２’であり、プログラムＣの実行カウンタが‘１’である。

これより、まず、「変換後の定周期プログラムＡ」におけるステップＳ３の判定はＮＯとなるので（位相定義値＝２なので）、ステップＳ４の処理が実行されることなく、ステップＳ５が実行される（実行カウンタ＝３となる）。

【０１００】

また、「変換後の定周期プログラムＢ」におけるステップＳ３の判定はＹＥＳとなるので（位相定義値＝２なので）、ステップＳ４の処理が実行される（定周期プログラムＢの実質的な実行が成される；よって星印（☆）がある）。そして、ステップＳ５で実行カウンタが＋１インクリメントされる（実行カウンタ＝３となる）。

【０１０１】

「変換後の定周期プログラムＣ」の実行に関しては、まず、ステップＳ１の判定がＹＥＳとなるので（周期定義値＝１なので）、実行カウンタは０クリアされる。これより、ステップＳ３の判定はＹＥＳとなるので（位相定義値＝０なので）、ステップＳ４の処理が実行され（定周期プログラムＣの実質的な実行が成される；よって星印（☆）がある）、ステップＳ５が実行される（実行カウンタ＝１となる）。

【０１０２】

経過時間＝５００（ｍｓ）以降の処理については説明を省略するが、８００（ms）経過時点で、上記０（ms）のときと同じく、全定周期プログラム（特にプログラムＡ，Ｂ）の実行カウンタは‘０’にリセットされる。つまり、プログラムＡ，Ｂに関しては、実質的に８００（ms）周期で動作することになる。そして、図９（ｂ）に示すように、プログラムＡに関しては実質的な実行周期タイミング（０（ms）時点）から２００（ms）後に実行されることになり、これは実質的に位相（２００（ms））が反映された形となる。プログラムＢに関しても同様に、実質的な実行周期タイミング（０（ms）時点）から４００（ms）後に実行されることになり、これは実質的に位相（４００（ms））が反映された形となる。

【０１０３】

上述したように、ＰＬＣ２の場合、位相の設定があっても、そのＯＳ処理自体では位相の設定内容は反映されないが、本手法によって、実行周期が上記最大公約数へと変更されており且つ「変換後の定周期プログラム」を実行することによって、実質的に、位相の設定内容が反映される形で各定周期プログラムの実質的な実行タイミングが決定される。つまり、ＰＬＣ２のようなＰＬＣで実行される場合であっても、図９（ｂ）に示すように、例えば図１１（ｂ）に示す例と同様のタイミングで各定周期プログラムが実行されるようになる。

【０１０４】

以上説明したように、本例の支援装置３では、“間欠実行機能を持たないＰＬＣ”（例えば上記IEC61131-3標準に対応したＰＬＣ等）であるＰＬＣ２で各定周期プログラムを間欠実行させる為に、各定周期プログラムを下記のように変換する。

【０１０５】

すなわち、まず、間欠実行させる定周期プログラム全ての実行周期および位相の最大公約数となる時間を求める。そして、この最大公約数を元に、実行周期、位相の約数を算出する。例えば、各定周期プログラム毎に、その実行周期、位相を、各々、最大公約数で除

算することで得られる各算出値は、実行周期、位相各々の約数の１つと言える（最大公約数に応じた約数と言える）。

【０１０６】

そして、各定周期プログラム毎に、その“最大公約数に応じた約数”を用いて、「変換後の定周期プログラム」を生成する。「変換後の定周期プログラム」は、その定周期プログラムに係わる上記“最大公約数に応じた約数”に基づく所定の条件を満たす場合に、その定周期プログラムが実行されるようにするものである。その為の一例が、上記図３に示す「変換後の定周期プログラム」のフォーマット（雛形）である。

【０１０７】

図３の例では、「変換後の定周期プログラム」の実行回数（実行カウンタ値）と、“最大公約数に応じた約数”とに基づいて、その定周期プログラムの実質的な実行タイミング（ステップＳ４の処理を実行すること）が決定される。実行カウンタ値と“最大公約数に応じた実行周期の約数”（＝実行周期÷最大公約数）との比較判定によって、上記実行周期タイミングを判定する。実行周期タイミングとなる毎に、実行カウンタ値を‘０’リセットする。そして、実行周期タイミングから位相分の時間経過したことを、実行カウンタ値と“最大公約数に応じた位相の約数”（＝位相÷最大公約数）との比較判定によって判定する。これによって、実行周期タイミングから位相分の時間経過時に、その定周期プログラムの実質的な実行（ステップＳ４の処理）が行われることになる。

【０１０８】

尚、上記“約数”を用いる例は、ＰＬＣ２側においては「変換後の定周期プログラム」を上記最大公約数の周期で実行させることを前提としている。但し、この例に限らない。

尚、図３に示すように、実行カウンタ値は、「変換後の定周期プログラム」の実行毎に＋１インクリメントされる。

【０１０９】

ＰＬＣ２においては、全ての「変換後の定周期プログラム」を、上記最大公約数の周期でそれぞれ実行させる。これは、例えば、ＰＬＣ２側で保持するタスクテーブル５０の実行周期５２を、全て、上記最大公約数に書き換えることで実現するが、この例に限らない。

【０１１０】

上記のように各定周期プログラムは、上記最大公約数の時間周期で起動が行われるが、「変換後の定周期プログラム」に含まれる元々の定周期プログラムは、実質的に実行周期タイミングから位相分の時間経過したタイミングになるまで実行されない。つまり、実質的に、定周期プログラムを位相分ズラして実行させることができる（つまり、各定周期プログラムの実質的な実行タイミングが出来るだけ重ならないようにでき、以って定周期性を守ることが出来るようになる）。

【０１１１】

以上説明したように、本例のプログラマブルコントローラシステム１（その支援装置３、ＰＬＣ２など）は、例えば位相制御機能を持つＰＬＣ用に作成されていた複数の定周期プログラムを、位相制御機能を持たないＰＬＣ２（例えばIEC61131-3標準に対応したＰＬＣ）で問題なく実行できる（定周期性を守ることが出来る）ように、当該“定周期プログラムを「変換」（または「補完」）”するものである（但し、更に、実行周期５２を上記最大公約数となるように補正する必要がある）。変換後（補完後）の定周期プログラムの一例を図６（ａ）に示しているが、この例に限らない。

【０１１２】

この様に、本例のプログラマブルコントローラシステム１（その支援装置３、ＰＬＣ２など）によれば、位相制御機能を持たないＰＬＣ２（例えばIEC61131-3標準に対応したＰＬＣ）であっても、定周期プログラムの定周期性を守ることが出来る。

【０１１３】

本例のプログラマブルコントローラシステム１（その支援装置３、ＰＬＣ２など）によれば、例えば位相制御機能を持つＰＬＣ用に作成されていた複数の定周期プログラムを、

位相制御機能を持たないＰＬＣに、容易に（手間が掛からない）移植することができる。

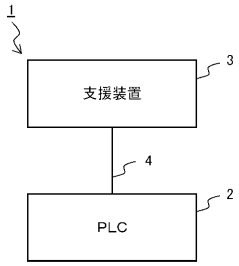
【符号の説明】

【０１１４】

１　ＰＬＣシステム
２　ＰＬＣ
３　支援装置
４　通信線
１１　変換元プログラム格納部
１２　実行周期・位相定義情報格納部
１３　プログラム変換部
１４　変換先プログラム格納部
１５　プログラム転送部
２１　制御プログラム実行部
３０　位相・周期定義表
３１　タスク
３２　実行周期
３３　位相
５０　補正後のタスクテーブル」
５１　タスク
５２　実行周期
５３　位相
５４　時間

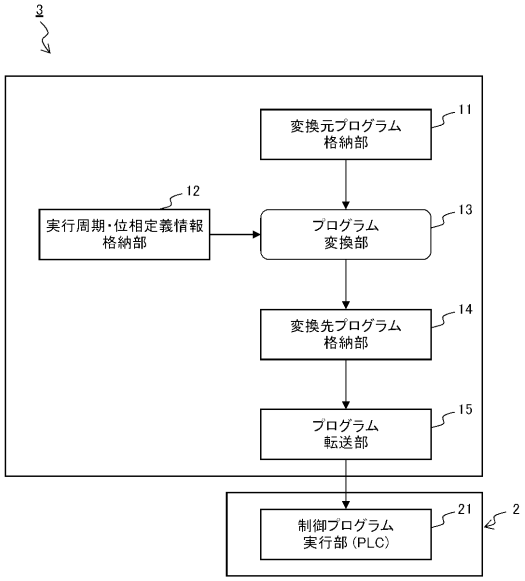
【図１】

プログラマブルコントローラシステムの概略構成図



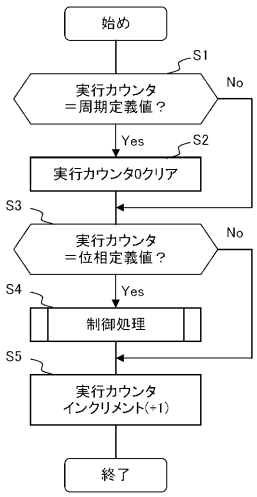
【図２】

本例の支援装置の機能ブロック図



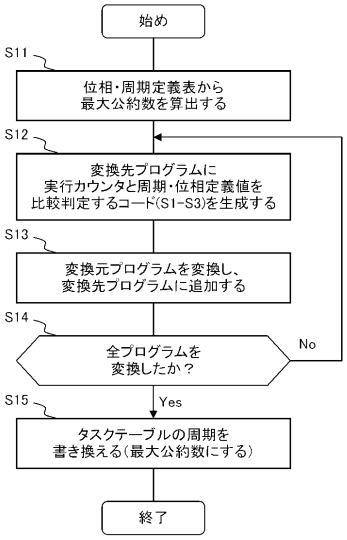
【図３】

「変換後の定周期プログラム」の雛形（フォーマット）の例



【図４】

プログラム変換部の処理フローチャート図



【図５】

（a）、（b）は、位相・周期定義表の具体例（その１）、（その２）

30

（a）

31 タスク	32 実行周期	33 位相
A	800(ms)	200(ms)
B	800(ms)	400(ms)
C	200(ms)	0(ms)

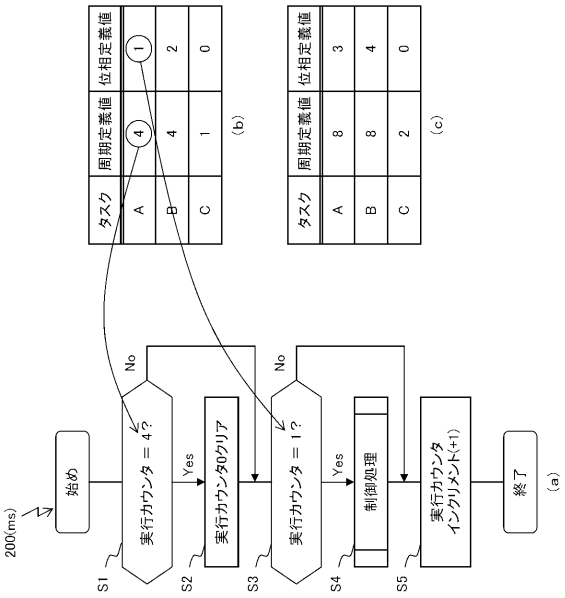
30

（b）

31 タスク	32 実行周期	33 位相
A	800(ms)	300(ms)
B	800(ms)	400(ms)
C	200(ms)	0(ms)

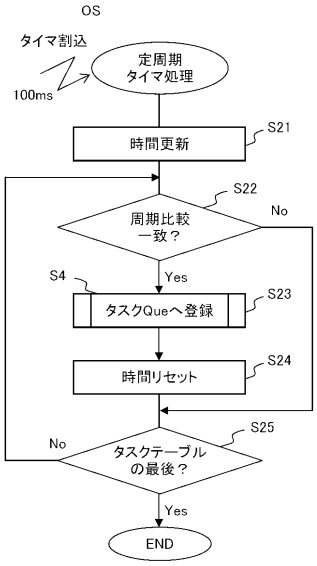
【図６】

（b）、（c）は周期定義値、位相定義値の具体例（その１）、（その２）であり、
（a）は「変換後の定周期プログラム」の具体例



【図７】

PLCのOSの処理フローチャート図



【図８】

「補正後のタスクテーブル」の一例

50

51 タスク	52 実行周期	53 位相	54 時間
A	200(ms)	200(ms)	—
B	200(ms)	400(ms)	—
C	200(ms)	0(ms)	—

【図９】

(a)、(b)は、処理実行の有無等に係わる具体例

(a)

経過時間(ms) / タスク	0 (800)	100	200	300	400	500	600	700	800
A	200 1 0 △	100	200 1 0 △	100	200 1 0 △	100	200 1 0 △	100	200 1 0 △
B	200 1 0 △	100	200 1 0 △	100	200 1 0 △	100	200 1 0 △	100	200 1 0 △
C	200 1 0 △	100	200 1 0 △	100	200 1 0 △	100	200 1 0 △	100	200 1 0 △

(b)

経過時間(ms) / タスク	0 (800)	100	200	300	400	500	600	700	800
A	4→0→1	1	1→2 ☆	2	2→3	3	3→4	4	4→0→1
B	4→0→1	1	1→2	2	2→3 ☆	3	3→4	4	4→0→1
C	1→0→1 ☆	1	1→0→1 ☆	1	1→0→1 ☆	1	1→0→1 ☆	1	1→0→1 ☆

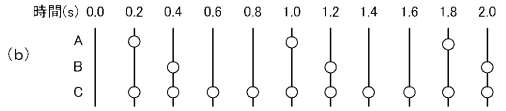
【図１１】

(a), (b)は、従来において位相を用いて実行タイミングをズラす例

(a)

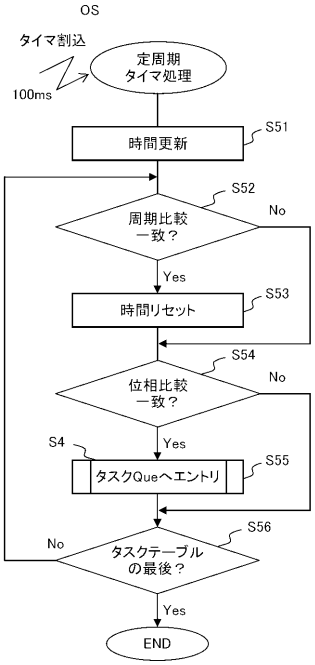
プログラム(タスク)	位相	周期
A	0.2s	0.8s
B	0.4s	0.8s
C	0.2s	0.2s

(b)



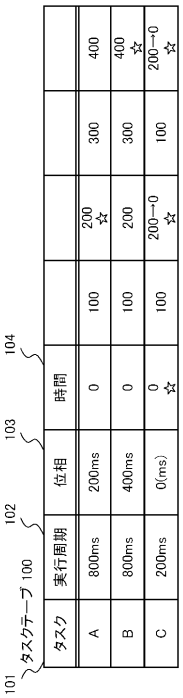
【図１２】

従来のPLCのOSの処理フローチャート図



【図１３】

従来のPLC側で保持するタスクテーブルの一例

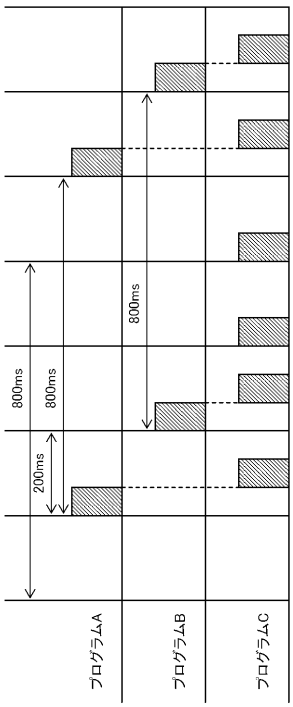


タスク	実行周期	位相	時間			
A	800ms	200ms	0	100	200 ☆	300	400	400 ☆	200→0 ☆
B	800ms	400ms	0	100	200	300	400 ☆		
C	200ms	0(ms)	0 ☆	100	200→0 ☆	100			

タスクテーブル 100
101
102
103
104

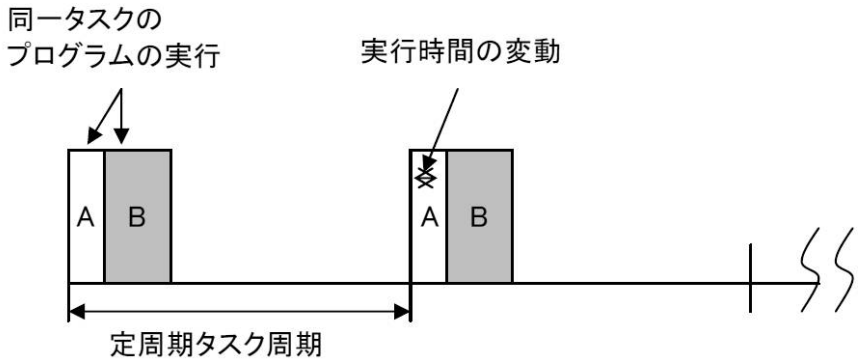
【図１４】

従来のプログラム実行タイミング例

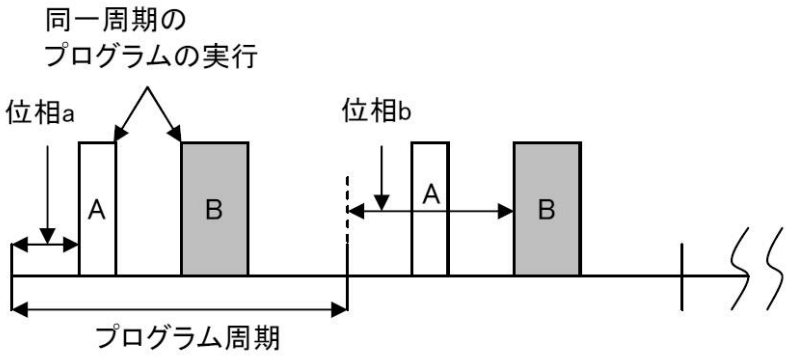


【図１０】

（a）、（b）は、実行周期が同じプログラムの実行処理例



（a）



（b）