

(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2012-185671
(P2012-185671A)

(43) 公開日 平成24年9月27日 (2012.9.27)

(51) Int.Cl.	F I	テーマコード (参考)
G 0 6 F 11/34 (2006.01)	G O 6 F 11/34 P	5 B O 4 2
G 0 6 F 9/48 (2006.01)	G O 6 F 9/46 4 5 2 J	
G 0 6 F 11/30 (2006.01)	G O 6 F 11/30 3 1 O A	
	G O 6 F 11/34 S	

審査請求 未請求 請求項の数 4 O L (全 12 頁)

(21) 出願番号	特願2011-48223 (P2011-48223)	(71) 出願人	000004260
(22) 出願日	平成23年3月4日 (2011.3.4)		株式会社デンソー
			愛知県刈谷市昭和町1丁目1番地
		(74) 代理人	110000578
			名古屋国際特許業務法人
		(72) 発明者	佐田 直樹
			愛知県刈谷市昭和町1丁目1番地 株式会
			社デンソー内
		Fターム(参考)	5B042 GA23 GB08 JJ20 JJ29 KK07
			LA20 MC25 MC29

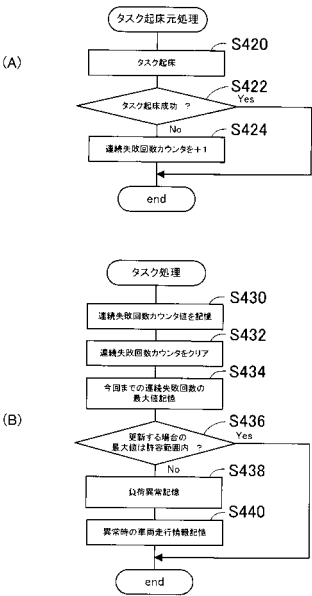
(54) 【発明の名称】 電子制御装置

(57) 【要約】

【課題】ソフトウェアの処理負荷が所定負荷を超えた場合に、処理負荷の異常原因を解析できる電子制御装置を提供する。

【解決手段】タスクの起床元では、起床対象のタスクを起床すると（S420）、起床対象のタスクからリターンされる起床結果に基づいて起床が成功したか否かを判定し（S422）、起床に失敗すると（S422：No）、連続失敗回数カウンタを＋1する（S424）。起床されたタスクは、起床元のタスクでカウントした連続失敗回数カウンタの値を今回値として記憶し（S430）、連続失敗回数カウンタをクリアし（S432）、今回の起床までにおける連続失敗回数の最大値を算出する（S434）。起床されたタスクは、連続失敗回数がソフトウェアの設計時に設定した許容範囲を超えている場合（S436：No）、ソフトウェアの処理負荷異常と判定し、判定結果と処理負荷異常時の車両走行情報とを記憶する（S438、S440）。

【選択図】 図4



【特許請求の範囲】

【請求項 1】

車両に搭載される電子制御装置において、
ソフトウェアの処理負荷を検出する負荷検出手段と、
前記負荷検出手段が検出した前記処理負荷が予め設定された所定負荷を超えると、前記ソフトウェアの処理負荷異常と判定する異常判定手段と、
前記異常判定手段が前記処理負荷異常と判定すると、前記処理負荷異常の判定結果と車両走行情報とを記憶部に記憶する異常記憶手段と、
を備えることを特徴とする電子制御装置。

【請求項 2】

前記負荷検出手段は、前記処理負荷として、同じタスクが起床されるときに前記タスクの起床が連続して失敗する連続失敗回数を検出し、
前記異常判定手段は、前記連続失敗回数が所定回数を超えると前記処理負荷異常であると判定する、
ことを特徴とする請求項 1 に記載の電子制御装置。

【請求項 3】

前記負荷検出手段は、前記処理負荷として、同じタスクが起床される起床回数と前記タスクの起床が失敗する起床失敗回数とを検出し、
前記異常判定手段は、前記起床回数に対する前記起床失敗回数の比率である起床失敗率が所定比率を超えると前記処理負荷異常であると判定する、
ことを特徴とする請求項 1 または 2 に記載の電子制御装置。

【請求項 4】

前記負荷検出手段は、前記処理負荷として、タスクが起床されてから前記タスクの処理が完了するまでに要する処理時間を検出し、
前記異常判定手段は、前記処理時間が所定時間を超えると前記処理負荷異常であると判定する、
ことを特徴とする請求項 1 から 3 のいずれか一項に記載の電子制御装置。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、車両に搭載される電子制御装置に関する。

【背景技術】

【0002】

従来、車両に搭載される電子制御装置において、制御部であるマイクロコンピュータ（マイコンとも言う。）が正常に作動しているか否かを監視するために、ウォッチドッグタイマ方式や宿題回答方式が採用されている。

【0003】

ウォッチドッグタイマ方式は、ウォッチドッグタイマがマイコンにより定期的にクリアされずにオーバーフローすると、マイコンの異常であると判定する。

また、宿題回答方式は、予め定められた監視用の信号をマイコンに送り、監視用信号に基づいてマイコンが所定の演算を行った演算結果をマイコンから受け取り、受け取った演算結果が正常値と異なる場合、あるいは所定時間内でマイコンから演算結果が回答されない場合、マイコンの異常であると判定する。

【0004】

このようなウォッチドッグタイマ方式や宿題回答方式を採用してマイコンの異常を監視しているときに所定の高負荷プログラムが実行されると、ウォッチドッグタイマのクリア処理や、宿題の回答処理が実行されないことがある。そのため、マイコンは正常であるにも関わらず、マイコンの異常と判定され、マイコンがリセットされるおそれがある。

【0005】

これに対し、特許文献 1 では、高負荷プログラムが実行されるときには、前述した宿題

10

20

30

40

50

回答方式による異常監視処理を停止させることにより、マイコンを異常と判定せず、マイコンをリセットしないようにしている。

【先行技術文献】

【特許文献】

【0006】

【特許文献1】特開2005-250854号公報

【発明の概要】

【発明が解決しようとする課題】

【0007】

マイコンにおいてソフトウェアの処理負荷が高くなり、前述したようなウォッチドッグタイマのクリア処理や、宿題の回答処理等が実行されない場合にマイコンがリセットされると、マイコンがリセットされた原因が処理負荷の上昇であることを検出できない。

【0008】

特許文献1においては、所定の高負荷プログラムが実行されると異常監視処理を停止するものの、異常監視処理を停止する対象である所定の高負荷プログラム以外のプログラムが実行され処理負荷が高くなった場合には、異常監視処理は停止されない。この場合には、マイコンから宿題が回答されないとしてマイコンがリセットされる。そのため、マイコンがリセットされた原因が処理負荷の上昇であることを検出できない。

【0009】

また、処理負荷が上昇したために車両制御上必要な処理が実行されなかったり待ち合わされたりすると、車両挙動に異常が発生することがある。この場合、処理負荷が上昇したためにマイコンがリセットされると、処理負荷の上昇が原因で車両挙動の異常が発生したことを検出できない。

【0010】

本発明は、上記課題を解決するためになされたものであり、ソフトウェアの処理負荷が所定負荷を超えた場合に、処理負荷の異常原因を解析できる電子制御装置を提供することを目的とする。

【課題を解決するための手段】

【0011】

請求項1から4に記載の発明によると、車両に搭載される電子制御装置において、ソフトウェアの処理負荷を検出する負荷検出手段が検出した処理負荷が予め設定された所定負荷を超えることにより、異常判定手段が処理負荷異常と判定すると、異常記憶手段は処理負荷異常の判定結果と車両走行情報とを記憶部に記憶する。

【0012】

この構成によれば、定期点検または車両の挙動異常時の点検等において処理負荷異常の判定結果を記憶部から読み出すことにより、処理負荷が所定負荷を超えたことを検出できる。さらに、処理負荷異常時の車両走行情報を記憶部から読み出すことにより、処理負荷が所定負荷を超えて処理負荷異常となった原因を車両走行情報に基づいて解析できる。

【0013】

ソフトウェアの処理負荷異常であることを判定する場合、請求項2に記載の発明のように、同じタスクが起床されるときにタスクの起床が連続して失敗する連続失敗回数を検出し、連続失敗回数が所定回数を超えると処理負荷異常であると判定してもよい。

【0014】

タスクの起床失敗は、例えば、タスクを起床させるときに、前回起床されたタスクの処理が完了しておらず、まだ実行中であるときに発生する。タスクを起床させるときに該当するタスクよりも優先順位の高いタスクが実行中か、該当するタスクの処理中に優先順位の高いタスクの処理が開始されると、該当するタスクの処理が待ち合わされて処理時間が延びる。その結果、タスクを起床させるときに前回起床されたタスクの処理が完了していない場合には、今回のタスクの起床が失敗し、タスクの処理が抜けることになる。

【0015】

10

20

30

40

50

そして、タスクの起床を連続して失敗する連続失敗回数が所定回数を超えることは、処理負荷が上昇したために該当するタスクの処理を所望の頻度で実行できないことを表わしている。この場合、処理負荷異常時に記憶部に記憶した車両走行情報に基づいて処理負荷異常の原因を解析することにより、処理負荷の異常上昇を避けるようにソフトウェアの仕様を修正できる。ソフトウェアの仕様修正として、例えば、タスクの起床条件、起床間隔、優先順位等の変更が考えられる。

【0016】

ソフトウェアの処理負荷異常であることを判定する場合、請求項3に記載の発明のように、同じタスクが起床される起床回数とタスクの起床が失敗する起床失敗回数とを検出し、起床回数に対する起床失敗回数の比率である起床失敗率が所定比率を超えると処理負荷異常であると判定してもよい。

10

【0017】

また、ソフトウェアの処理負荷異常であることを判定する場合、請求項4に記載の発明のように、タスクが起床されてからタスクの処理が完了するまでに要する処理時間を検出し、処理時間が所定時間を超えると処理負荷異常であると判定してもよい。

【0018】

起床失敗率が所定比率を超えるか、あるいはタスクの処理時間が所定時間を超えることは、処理負荷が上昇したために該当するタスクの処理を所望の頻度で実行できないことを表わしている。

【0019】

これらの場合にも、処理負荷異常時に記憶部に記憶した車両走行情報に基づいて処理負荷異常の原因を解析することにより、処理負荷の異常上昇を避けるようにソフトウェアの仕様を修正できる。

20

【図面の簡単な説明】

【0020】

【図1】第1実施形態による電子制御装置を示すブロック図。

【図2】車両点検形態を示す模式図。

【図3】車両点検の作業手順を示すフローチャート。

【図4】(A)はタスク起床元における異常検出処理を示すフローチャート、(B)は起床されるタスクにおける異常検出処理を示すフローチャート。

30

【図5】連続失敗回数に基づく異常検出を示すタイムチャート。

【図6】異常検出用の変数、異常判定結果および異常発生時の車両走行情報を記憶する記憶部の構成を示す模式図。

【図7】(A)は第2実施形態によるタスク起床元における異常検出処理を示すフローチャート、(B)は起床されるタスクにおける異常検出処理を示すフローチャート。

【図8】起床失敗率に基づく異常検出を示すタイムチャート。

【図9】(A)は第3実施形態によるタスク起床元における異常検出処理を示すフローチャート、(B)は起床されるタスクにおける異常検出処理を示すフローチャート。

【図10】タスクの処理時間に基づく異常検出を示すタイムチャート。

【発明を実施するための形態】

40

【0021】

以下、本発明の実施形態を図に基づいて説明する。

〔第1実施形態〕

図1に、車両に搭載される第1実施形態のエンジン用の電子制御装置(ECU: Electronic Control Unit)10を示す。エンジンECU(以下、単にECUとも言う。)10は、電源回路12、14、マイコン20、EEPROM40、入出力回路42等から構成されている。

【0022】

電源回路12は、スタートスイッチ4がオンになると、エンジンECU10の主な電子部品にバッテリー2の電力を供給する。電源回路14は、スタートスイッチ4のオン、オフ

50

に関わらず、常にエンジン ECU 10 の該当する電子部品、例えば S R A M (スタンバイ R A M) 28 にバッテリー 2 の電力を供給する。

【0023】

マイコン 20 は、C P U 22、R A M 24、R O M 26、S R A M 28、I / O 30 から主に構成されている。マイコン 20 は、R O M 26 に記憶されている制御プログラムを C P U 22 が実行することにより、A F センサ 50、回転数センサ 52、吸気温センサ 54、エアフロセンサ 56、水温センサ 58、車速センサ 60 等から車両走行状態を表わす検出信号を入力し、車両走行状態に基づいてエンジン制御を実行する。

【0024】

制御プログラムが作業用に使用し、スタートスイッチ 4 がオフになると記憶データが消失する R A M 24 に対し、S R A M 28 は、スタートスイッチ 4 のオン、オフに関わらず、電源回路 14 を介してバッテリー 2 から電力を供給されている。したがって、S R A M 28 に記憶されているデータは、バッテリー 2 の交換等によりバッテリー 2 からの電力供給が遮断されない限り保存される。

【0025】

I / O 30 は、入出力回路 42 を介して、各種センサから検出信号を入力したり、C A N 等の車内 L A N により他の E C U と通信する。

E E P R O M 40 は、マイコン 20 に外付けされている。E E P R O M 40 は、書き込み可能な不揮発性の記憶部であり、バッテリー 2 が交換されても、記憶されているデータは保存される。

【0026】

(車両点検)

定期点検や車両 100 の挙動異常時にユーザがディーラまたは修理工場に車両 100 を持ち込むと、図 2 に示すように、診断ツール 110 により車両 100 からダイアグコードおよび車両走行情報を読込んで異常原因が診断される。ディーラまたは修理工場とは別に、車両 100 の挙動異常が発生したときに、車両 100 から無線通信により情報センタ 120 にダイアグコードと車両走行情報を送信してもよい。

【0027】

(点検作業)

図 3 に、定期点検や車両の挙動異常時における点検作業の手順を示す。

定期点検または車両の挙動異常時にユーザがディーラ等に車両を持ち込むと (S 400)、ディーラでは、車両を点検するとともに、診断ツールにより車両からダイアグコードを読み出し、異常が発生している場合には、診断マニュアル等に基づいて適切な処置を施そうとする (S 402)。

【0028】

異常が発生している場合に適切な処置が実施され、異常が解決すると (S 404: Yes)、本実施形態の点検作業は終了する。

ダイアグコードからソフトウェアの処理負荷が異常原因であると判断されると (S 406: Yes)、ディーラではソフトウェアの処理負荷の異常原因を解析できないので、車両はソフトウェアの設計部門等に持ち込まれる。

【0029】

設計部門では、車両からソフトウェアの処理負荷異常時の車両走行情報を読み出し、処理負荷異常の原因を解析する (S 408)。ソフトウェアの処理負荷異常時の車両走行情報は、異常発生時に E C U 10 により記憶されている。

【0030】

ソフトウェアの処理負荷異常の原因を解析した結果、処理負荷異常の原因がソフトウェアの制御仕様の場合 (S 410: Yes)、ソフトウェアの制御仕様の修正要と判断する (S 412)。そして、処理負荷異常を解消するために、ソフトウェアの制御仕様が検討される。

【0031】

10

20

30

40

50

(処理負荷の異常検出処理)

E C U 1 0においては、エンジン制御のために複数のタスクが実行される。各タスクには、起床条件、起床間隔、優先順位等が設定されている。複数のタスクが実行されているときにソフトウェアの処理負荷が上昇し、例えば優先順位の高いタスクの実行頻度が増えると、優先順位の低いタスクは、優先順位の高いタスクの処理が完了するまで、処理が待ち合わせされる。

【0032】

タスクの処理が待ち合わせされると、タスクが処理する車両制御の頻度が低くなったり、処理完了までの処理時間が長くなったりする。その結果、車両制御に不具合が生じ、車両挙動に異常が発生することがある。

【0033】

図4に、E C U 1 0において実行されるソフトウェアの処理負荷異常の検出処理を示す。図4の(A)はタスクの起床元での処理を示し、図4の(B)は起床される側のタスクの処理を示している。

【0034】

タスクの起床元では、図4の(A)に示すように、タスクを起床すると(S420)、起床対象のタスクからリターンされる起床結果に基づいて、起床が成功したか否かを判定する(S422)。起床に成功した場合には(S422:Yes)、本処理を終了する。

【0035】

ここで、図5に、優先順位の低い低タスクと、それよりも優先順位の高い高タスクとが起床されるときに、同じ低タスクの起床が連続して失敗する連続失敗回数の変化を示す。図5では、起床成功、起床失敗を含め13回低タスクが起床されている。1回目～3回目の起床は成功しているので、連続失敗回数カウンタの値は0のままである。3回目の起床時には高タスクが起床されるので低タスクの処理開始は待ち合わせされているが、4回目の起床前に高タスクの処理が完了し低タスクの処理が開始されるので、3回目の起床は成功になっている。

【0036】

一方、図5の4回目、5回目の起床時には、3回目の起床で処理を開始した低タスクの処理が完了していないので、起床時に起床失敗になっている。

このように起床に失敗すると(S422:No)、起床元のタスクは、R A M 2 4 (図6参照)に設定された連続失敗回数カウンタを+1する(S424)。第1実施形態では、ソフトウェアの設計時に設定した連続失敗回数の許容範囲を表わす所定回数を「2回」にしている。したがって、連続失敗回数が2回以下であれば正常であり、2回を超えると異常になる。

【0037】

尚、複数のタスクを異常検出処理の対象として連続失敗回数をカウントする場合、異常検出処理の対象タスク毎に連続失敗回数カウンタが設定される。

図5では、1回目～3回目の起床において低タスクの起床が成功しているので、連続失敗回数カウンタの値は0のままである。4回目、5回目の起床は失敗になっており、後述するように起床されたタスクにより連続失敗回数カウンタがクリアされないので、連続失敗回数カウンタがそれぞれ+1され、1、2と増加している。

【0038】

タスクの起床が成功すると、図4の(B)に示すように、起床されたタスクは、起床元のタスクでカウントした連続失敗回数カウンタの値を今回値としてR A M 2 4に記憶し(S430)、連続失敗回数カウンタをクリアする(S432)。そして、連続失敗回数の今回値と、R A M 2 4に記憶されている前回までの連続失敗回数の最大値とを比較し、大きい方の値を連続失敗回数の最大値として記憶する(S434)。

【0039】

図5の6回目の起床では、高タスクの処理完了後に低タスクの処理が開始しているので、起床は成功している。したがって、起床された低タスクは、連続失敗回数カウンタの値

10

20

30

40

50

である「2」を今回値としてRAM 24に記憶し、連続失敗回数カウンタをクリアして0にする。連続失敗回数の最大値の初期値は0であるから、連続失敗回数の今回値の「2」が最大値として記憶される。

【0040】

第1実施形態では、連続失敗回数の許容範囲を2回以下としているので、今回更新した連続失敗回数の最大値の「2」は許容範囲内である。最大値が許容範囲内であれば（S 436：Yes）、本処理を終了する。

【0041】

図5の9回目～11回目の起床時には、8回目の起床で処理を開始した低タスクの処理が完了していないので、起床時に起床失敗になっている。したがって、11回目の起床時において連続失敗回数のカウント値は「3」に設定される。

10

【0042】

9回目～11回目の3回の起床に失敗した後、12回目の起床は成功するので、連続失敗回数の最大値は「3」に更新される。12回目の起床により処理を開始した低タスクでは、更新された連続失敗回数の最大値の「3」は許容範囲を超えており、ソフトウェアの処理負荷異常であると判定する（S 436：No）。

【0043】

この場合、起床されたタスクは、ソフトウェアの処理負荷異常と判定した判定結果と処理負荷異常時の車両走行情報とをSRAM 28またはEEPROM 40に記憶し（S 438、S 440）、処理負荷異常が発生したことを通知する。ダイアグタスクは、ソフトウェアの処理負荷異常が通知されると、該当するダイアグコードを設定する。

20

【0044】

SRAM 28またはEEPROM 40に記憶する車両走行情報として、吸気量、吸気温、水温、エンジン回転数、車速、連続失敗回数が所定回数を超えたタスク名等が考えられる。車両走行情報としては、車両自体の走行情報だけでなく、ナビゲーション装置等から取得する道路の曲率、勾配等の道路形状、天気、渋滞等の車両周囲の環境を表わす走行情報を記憶してもよい。

【0045】

尚、ソフトウェアの処理負荷異常と判定した判定結果と処理負荷異常時の車両走行情報とが記憶される記憶部は、車両のスタートスイッチがオフになってもデータが保持されるものであれば本実施形態で例示したSRAM（スタンバイRAM）またはEEPROMに限るものではなく、EEPROM以外の書き換え可能な不揮発性メモリであってもよい。

30

【0046】

第1実施形態では、ECU 10が本発明の電子制御装置に相当し、SRAM 28またはEEPROM 40が本発明の記憶部に相当する。そして、ECU 10は、負荷検出手段、異常判定手段および異常記憶手段が実行する機能を実現する。

【0047】

また、図4のS 422、S 424、S 430～S 434の処理が本発明の負荷検出手段が実行する機能に相当し、S 436の処理が本発明の異常判定手段が実行する機能に相当し、S 438およびS 440の処理が本発明の異常記憶手段が実行する機能に相当する。

40

【0048】

[第2実施形態]

本発明の第2実施形態による処理負荷の異常検出処理を図7および図8に示す。第2実施形態の異常検出処理は、起床失敗率に基づいて処理負荷異常を検出する。第2実施形態では、ソフトウェアの設計時に設定した起床失敗率の許容範囲を表わす所定比率を「50%」にしている。したがって、起床失敗率が50%以下であれば正常であり、50%を超えると異常になる。

【0049】

図7の（A）に示すように、起床元のタスクでは、低タスクを起床すると（S 450）、起床の成功、失敗に関わらず所定回数まで起床回数をカウントアップする（S 452）

50

。

【0050】

一方、図7の(B)に示すように、起床に成功して低タスクの処理が開始すると、低タスクでは、起床されて処理が開始される毎に処理回数カウンタをカウントアップする(S470)。図8に示すように、1回目～3回目の起床は成功しているが、4回目および5回目の起床は、3回目の起床により処理を開始した低タスクの処理が完了していないために失敗しているので、処理回数カウンタの値は3回そのまま更新されていない。

【0051】

尚、複数のタスクを異常検出処理の対象として起床回数および処理回数をカウントする場合、異常検出処理の対象タスク毎に起床回数カウンタ、処理回数カウンタが設定される

10

。

【0052】

起床元タスクでは、起床回数が所定回数以上になると(S454:Yes)、次式(1)に基づいて起床失敗率を算出し(S456)、RAM24に起床失敗率を記憶する。そして、処理回数カウンタおよび起床回数カウンタをクリアする(S460)。

【0053】

起床失敗率 [%]

$$= \{ (\text{起床回数カウンタ} - \text{処理回数カウンタ}) / \text{起床回数カウンタ} \} \times 100$$

・・・(1)

第2実施形態では、所定回数は100回であり、ソフトウェアの設計時に設定する起床失敗率の許容範囲は前述したように50%以下である。図8の起床結果に基づいて、起床回数カウンタ=100、処理回数カウンタ=46を式(1)に代入して起床失敗率を算出すると、起床失敗率=54%になる。

20

【0054】

S456で算出した所定回数(100回)の起床処理までの起床失敗率がソフトウェアの設計時に設定した許容範囲(第2実施形態では50%以下)を超えている場合(S462:No)、起床元タスクは、ソフトウェアの処理負荷異常と判断する。

【0055】

そして、起床元タスクは、ソフトウェアの処理負荷異常と判定した判定結果と処理負荷異常時の車両走行情報とをSRAM28またはEEPROM40に記憶し(S464、S466)、処理負荷異常が発生したことを通知する。ダイアグタスクは、ソフトウェアの処理負荷異常が通知されると、該当するダイアグコードを設定する。

30

【0056】

第2実施形態では、図7のS452～S456およびS470の処理が本発明の負荷検出手段が実行する機能に相当し、S462の処理が本発明の異常判定手段が実行する機能に相当し、S464およびS466の処理が本発明の異常記憶手段が実行する機能に相当する。

【0057】

[第3実施形態]

本発明の第3実施形態による処理負荷の異常検出処理を図9および図10に示す。第3実施形態の異常検出処理は、タスクが起床されてからタスクの処理が完了するまでの処理時間に基づいて処理負荷異常を検出する。第3実施形態では、ソフトウェアの設計時に設定した処理時間の許容範囲を表わす所定時間を「4」にしている。したがって、処理時間が4以下であれば正常であり、4を超えると異常になる。

40

【0058】

図9の(A)に示すように、起床元のタスクでは、低タスクを起床し(S480)、起床時刻をRAM24に記憶する(S482)。例えば、図10において、1回目～4回目の起床時刻は、「1」、「8」、「15」、「22」である。

【0059】

図9の(B)に示すように、起床されたタスクは、タスクとして割り当てられた処理を

50

実行すると（S 4 9 0）、処理完了時刻をR A M 2 4に記憶する（S 4 9 2）。起床元タスクで記憶した起床時刻とS 4 9 2で記憶した処理完了時刻とから、起床されたタスクの処理時間を算出し（S 4 9 4）、処理時間がソフトウェアの設計時に設定した許容範囲内であるか否かを判定する（S 4 9 6）。

【0060】

尚、複数のタスクを異常検出処理の対象とする場合、異常検出処理の対象タスク毎に処理時間が算出される。

図10において、1回目～4回目の起床における低タスクの処理完了時刻は、「3」、「11」、「20」、「24」である。したがって、1回目～4回目の起床における各低タスクの処理時間は、処理完了時刻と起床時刻との差から、「2」、「3」、「5」、「2」と算出される。図10においては、処理時間が「4」を超えると、処理時間が許容範囲を超えていると判断している。

10

【0061】

処理時間が許容範囲（第3実施形態では4以下）を超えると（S 4 9 6：No）、起床されたタスクは、ソフトウェアの処理負荷異常と判断し、ソフトウェアの処理負荷異常と判定した判定結果と処理負荷異常時の車両走行情報とをS R A M 2 8またはE E P R O M 4 0に記憶し（S 4 9 8、S 5 0 0）、処理負荷異常が発生したことを通知する。ダイアグタスクは、ソフトウェアの処理負荷異常が通知されると、該当するダイアグコードを設定する。

【0062】

第3実施形態では、図9のS 4 8 2、S 4 9 2およびS 4 9 4の処理が本発明の負荷検出手段が実行する機能に相当し、S 4 9 6の処理が本発明の異常判定手段が実行する機能に相当し、S 4 9 8およびS 5 0 0の処理が本発明の異常記憶手段が実行する機能に相当する。

20

【0063】

以上説明した上記複数の実施形態によると、同じタスクを起床するときに連続して起床に失敗する連続失敗回数、同じタスクが起床された起床回数に対する起床に失敗した起床失敗回数の比率、あるいはタスクが起床されてからタスクの処理が完了するまでに要する処理時間のいずれかがソフトウェアの設計時に設定した許容範囲をこえると、ソフトウェアの処理負荷の異常と判断する。

30

【0064】

そして、処理負荷の異常発生時の判定結果を表わすダイアグコードと異常発生時における車両走行情報とを、車両のスタートスイッチがオフになっても記憶データが消失しないS R A M 2 8またはE E P R O M 4 0に記憶する。

【0065】

これにより、定期点検または車両の挙動異常時に診断ツールでダイアグコードを読み出せば、ソフトウェアの処理負荷異常が発生していた場合に、処理負荷異常を検出できる。そして、処理負荷異常発生時の車両走行情報に基づいて処理負荷の異常原因を解析することにより、ソフトウェアの処理負荷の異常が発生しないように、ソフトウェアの仕様を修正する等の適切な処置を実施することができる。

40

【0066】

また、上記実施形態で説明したソフトウェアの処理負荷異常の検出処理をソフトウェアの開発段階で実行すれば、ソフトウェアの処理負荷が所定負荷を超えないように、ソフトウェアの開発段階で仕様を設定できる。

【0067】

〔他の実施形態〕

上記実施形態では、連続失敗回数、起床失敗率または処理時間がソフトウェアの設計時に予め設定した所定値を超えると、ソフトウェアの処理負荷異常と判定した。これ以外にも、所定期間中にアイドルタスクが実行される時間の比率であるアイドル率が所定値を下回ると、ソフトウェアの処理負荷異常と判定してもよい。

50

【0068】

また、起床されたタスクの処理が完了してから同じタスクが起床されるまでの余裕時間が所定時間よりも短くなると、ソフトウェアの処理負荷異常と判定してもよい。

また、連続失敗回数、起床失敗率または処理時間を計測してソフトウェアの処理負荷異常の検出対象となるタスクは優先順位の低い低タスクに限るものではなく、優先順位に関わらず複数のタスクを検出対象とすることができる。

【0069】

また、異常検出対象のタスク毎に、連続失敗回数、起床失敗率および処理時間のうち2個以上を計測し、ソフトウェアの処理負荷が所定負荷を超えていることを計測値のいずれかが示せば、処理負荷異常と判定してもよい。

【0070】

上記実施形態では、異常判定結果および異常発生時の車両走行情報を、処理負荷異常の発生している同じECUの記憶部に記憶したが、別のECUの記憶部に記憶してもよい。

このように、本発明は、上記実施形態に限定されるものではなく、その要旨を逸脱しない範囲で種々の実施形態に適用可能である。

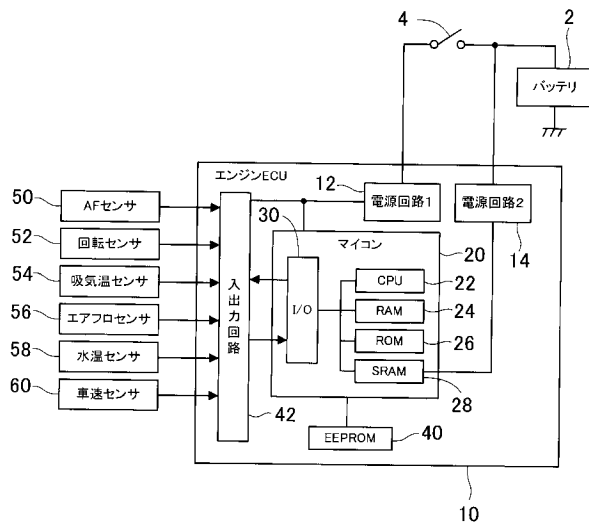
【符号の説明】

【0071】

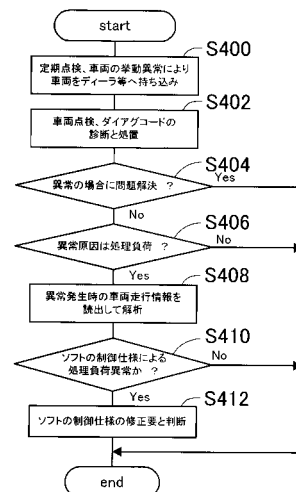
10：ECU（電子制御装置、負荷検出手段、異常判定手段、異常記憶手段）、SRAM 28（記憶部）、EEPROM 40（記憶部）

10

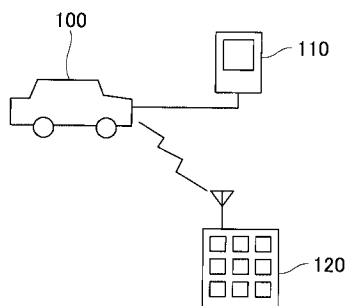
【図1】



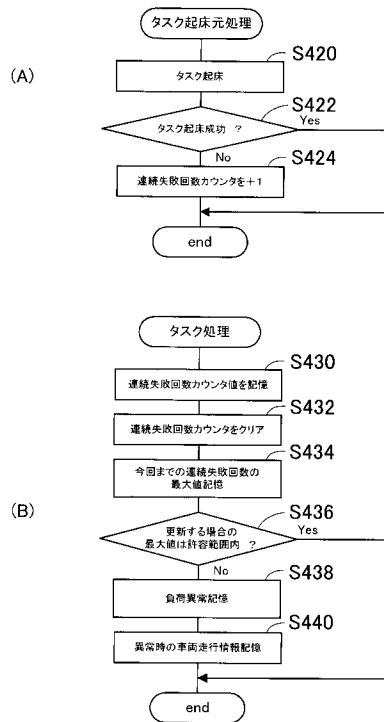
【図3】



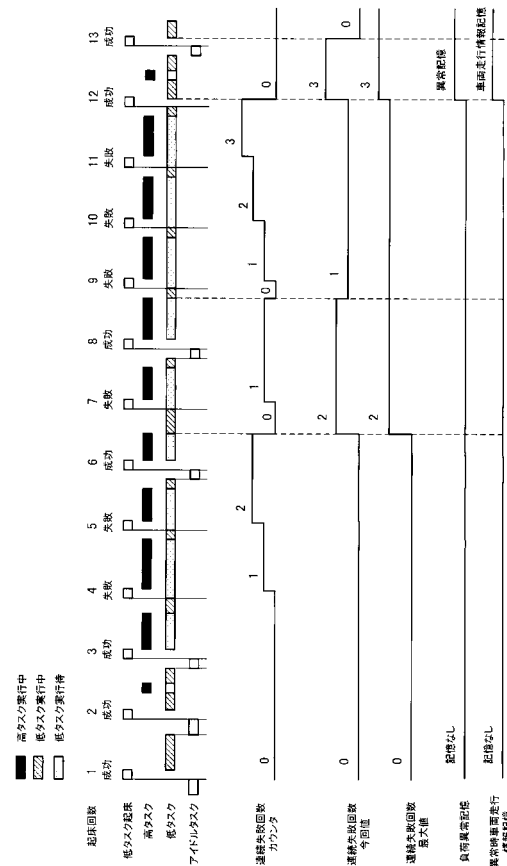
【図2】



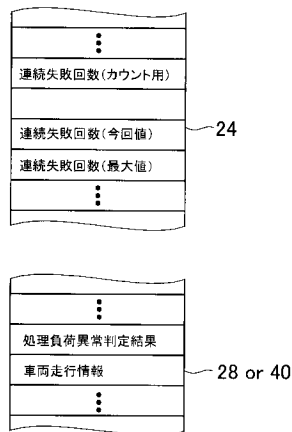
【図 4】



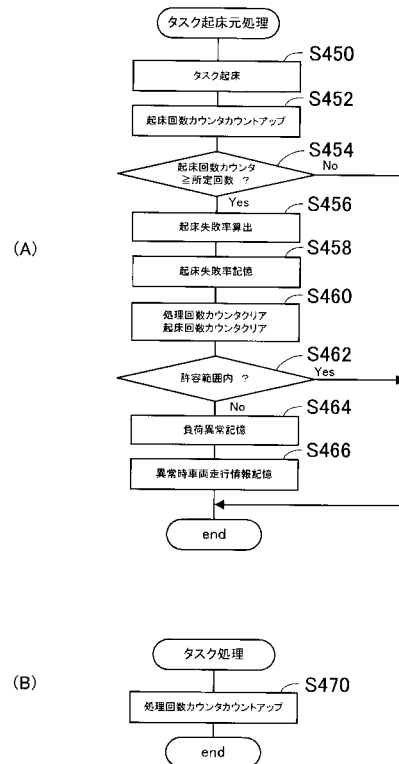
【図 5】



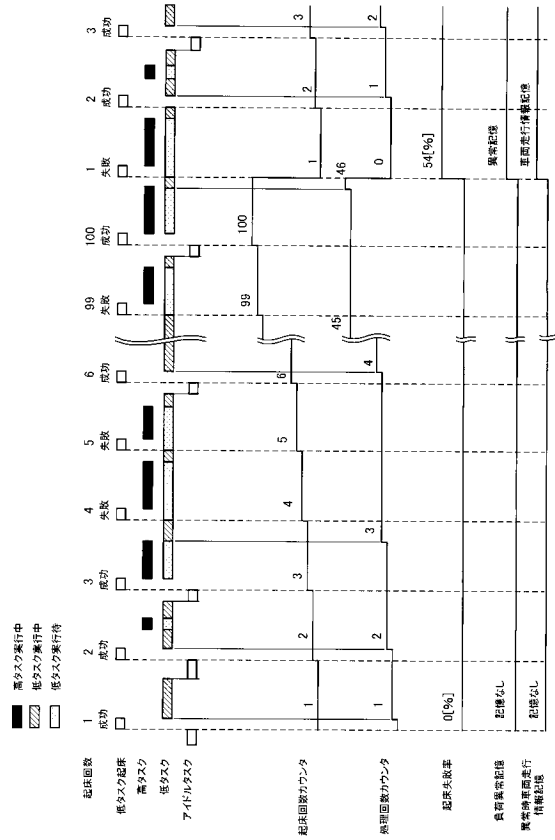
【図 6】



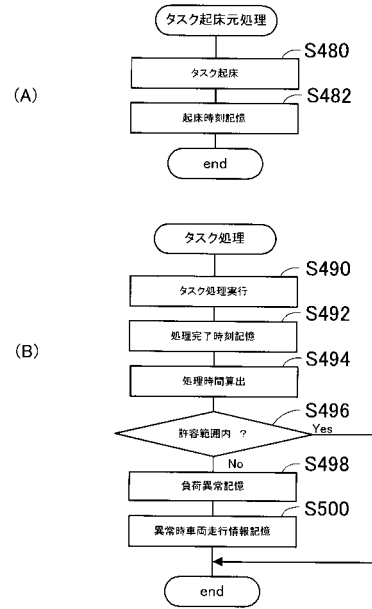
【図 7】



【図 8】



【図 9】



【図 10】

