UDC 681.3.326 : 629.113.52

エンジンコントロール用自動デバッグシステム

Engine Control Automatic Debug System

井手　忍[1]　八木　潔[2]　斗納宏敏[3]
Shinobu Ide　Kiyoshi Yagi　Hirotoshi Tono

要　旨

自動車の排ガス浄化・低燃費・安全性がより一層求められているため、マイコンによるエンジン制御の高機能化を推進してきた。また消費者の嗜好が多様化しているため、生産機種も多くなってきた。これらの理由により、プログラム設計量も増加の一途を辿っている。設計作業の中で最も工数を要するのは、デバッグである。設計容量が増大している中で、いかに、正確にデバッグを行いプログラムの信頼性を高めるか、また設計の効率化を図るかが重要な問題となってきたため、今回本システムの開発を行った。

デバッグを自動化するために、マイコンに入力する信号の全てを制御し、出力信号の形態に関わらず計測するようにしている。また設計条件をファイルに登録し、編集を行うようにしている。この他に、様々な機能を付加したことによって、信頼性の向上、設計工数の削減に効果を上げることができた。

As request for purer exhaust emission, fewer fuel consumption, and safer has been growing up, the engine control unit have been developped to have more funtion using the microprocessor. Further the diversification of consumer makes the number of production models incresing. Aboves oblige us to increase programs to be desinged. Among desing proces, debuggin needs longest time to work. Under such a circumstance as a huge capacity of program design, it is very important to make the program reliable through accurate debugging, and to improve design efficiency, which is the back ground of this development.

To automate the debugging, all input signals to the microprocessor are controlled and measuerd regardless the shape of the output signals. Also design conditions are registered and filed. Futher more adding various functions enables to improve the reliability and to reduce the design lead time.

(1)〜(3)　モートロニクス本部システム開発部

1. はじめに

地球規模による環境破壊や、省資源・省エネルギーの問題が大きく取り上げられており、自動車の排出ガスや燃費の規制は年々強化されてきた。このため、自動車はより一層の排ガス浄化・低燃費を図ることが要求されている。また、自動車事故の増加は深刻な社会問題の一つとなっているため、安全性の向上も重要な問題としてクローズアップされている。これらの理由から、エンジン制御の高機能化を推進してきた。例えば、燃料噴射制御・点火時期制御・アイドル回転数制御等のエンジンの基本的な制御や、これらの制御に異常が発生した時に作動するフェイルセーフ機能や、異常箇所の検出に役立つ自己診断機能等があげられる。これらの制御は通常マイクロコンピュータで行っており、機能を付加するほどプログラム容量は増大する。

また、かつては少品種を大量生産していた自動車産業であるが、最近では消費者の嗜好が多様化してきており、生産する種類が多品種に渡っている。このため、制御プログラムの種類も増大している。

これらの理由により、1人当たりの設計量も増加の一途を辿っているため、設計の効率化を図り、信頼性を向上するための方策が求められている。

2. 開発の目的

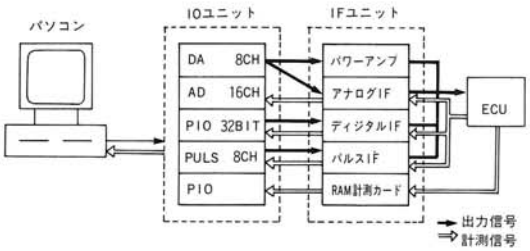
プログラムのデバッグは、設計したプログラムが要求仕様通りの処理をしているかを確認し、仕様に反した動作をしていれば、原因となる箇所を発見して修正をすることで行われる。この作業は要求仕様通りの動作を、満足するまで、何度となく繰り返すことになり、プログラム設計工数の中でも大きな比重を占めている。特に、エンジン制御における不具合は人命に関わる危険性があるため、プログラムの動作確認は厳密に行う必要があり、全工程の50%以上をデバッグに費やしている。

基本的なデバッグ作業は、仕様書に記述されている内容を一つ一つ確認することで行われるが、全ての条件下で要求仕様を確認することは非常に困難であり、作業量が膨大であるため、チェック洩れや見誤り、設定ミス等の人為的なミスが発生するおそれがある。プログラムの容量が増大し、機能が複雑化している上に設計量が増大している現状ではその可能性は高い。このため、いかに正確にデバッグを行い、プログラムの信頼性を高めるか、また設計の効率化を図るかが問題になってくる。

そこで、デバッグを自動化することにより、人為的ミスをなくし、プログラムの信頼性を向上させ、また、工程上大きなウエイトを占めるデバッグ作業の工数を削減することが必要となってきたため、本システム、ADS（Automatic Debug System = 自動デバッグ装置）の開発を行った。

プログラムのデバッグを支援するシステムとして、シミュレータとエミュレータがあげられる。シミュレータは主にソフトウェアのチェックのために用い、ハードウェアがなくてもプログラムのデバッグが可能である。但し、純粋にソフト上のチェックのみに止まり、実際のエンジン信号が入った時の動作に関しては、デバッグできない。エミュレータの一つであるICE（In-Circuit Emulater）は、実際のエンジンシステムに組み込んで評価を行うことができる。しかし、この場合もマイコン側から見た、ソフト中心のチェックとなり、周辺回路の信号設定を行う機能はない。自動車のエンジン制御プログラムのデバッグを自動化するため

図－1　ADSの外観
Fig. 1　Exterior veiw of ADS



図－2　ハード構成図
Fig. 2　Hardware structure

には、マイコンを含んだECU（Electronic Control Unit）に入力する信号全てを制御できる環境でチェックを行う必要がある。ADSは、ECUの入力条件の設定、CPUの内部情報（RAM）の設定、またECUが実際に出力する信号の計測を行うことができる。

3. 1　ADSシステムの概要

デバッグをするためには、ECUに自動車の状態を示すセンサ信号を入力しなければならない。このセンサ信号を様々に変えてデバッグしたい条件を作り出す。今までのデバッグは、例えば加速時とか始動時というそれぞれの条件において、デバッグの対象となる信号の操作を手動で行っていた。入力するセンサ信号の値を設定値通りにするのはなかなか困難な作業である上、デバッグする項目は多数にのぼり多大な時間を費やす。このため、自動化することにより入力信号の正確な調整を瞬時に行い、計測もスムーズに行うことができる。図－1にADSの外観をしめす。

3. 1. 1　ハード構成

ADSのハード構成は大きく分けて、次の3つから成っている。（図－2参照）

① システムのコントロール系（パソコン）

入力条件の設定、センサ信号の出力、入力信号の計測、自動計測等の動作を制御する装置。本システムではパーソナルコンピュータ、FM R-60HXを使用した。

② IO（Input/Output）ユニット・IF（Interface）ユニット

ECUに入力する信号の発生および出力信号の計測を行う装置。

③ ECU

デバッグの対象となる実際のECU。

エンジンを設定するセンサ信号は、アナログ・

表－1　IOユニット 各カードの特性

仕様 ＼ カード	D/A変換	A/D変換	PIO（デジタル信号入出力）
チャネル数	8CH	8CH	出力16CH　入力16CH
出力範囲	−5V～+5V	−10V～+10V	0V～+5V　32bit使用可能
分解能	12bit	12bit	
LSB	2.44mV	4.88mV	
カウンタクロック	1 μsec	1 μsec	

表－2　IFユニットの特性

カード	特性
アナログIF	＋B用20V出力 1CH，5V出力 7CH
デジタルIF	＋B，VCC出力切替スイッチ
パルスIF	分周回路（120℃A，180℃A，360℃A，720℃A）
RAM計測	指定アドレスのRAM値計測

パルス・デジタルの各種信号を使う。

IOユニットは、パソコンで設定したデータの入出力を行う。IFユニットは、IOユニットからの信号をエンジン信号の形に変換しECUに出力する。また、ECUから出力する信号を入力する装置である。

IOユニットは、アナログ信号・デジタル信号・パルス信号の入出力を行う。アナログ信号の入出力に関しては既成のDA変換カード・AD変換カードを、デジタル信号も既成のPIOカードを使用した。それぞれのカードの特性は表－1に記す。IFユニットの特性は表－2に記す。

パルスを表すセンサ信号およびECUから出力するパルスには様々な種類の形があるため、計測方法も多岐に渡る。従来の測定器を使って対応するには、カウンタやパルス信号発生器、位相計測器等のいろいろな種類のものを組み合わせて使用しなければならない。そこで今回新しく、一枚のカードだけで、自動車に使われている全てのパルス信号の発生・計測を行うことができるように専用のパルスカードを開発した。このカードの特徴は、ソフトの変更でパルス出力またはパルス計測の方法を自由に選択できることである。

パルスカードの詳細機能については表－3に記す。パルス出力は、ADSがECUに出力する信号で、例えば①は、エンジン回転数のセンサ信号を表した波形である。周期はソフト上で1μs～65536μsの間を指定する。パルス計測は、ECUが出力する制御信号で、例えば①は、インジェクタを開弁する時間（燃料を噴射する時間）を表した波形である。この、H（またはL）になっているパルス幅（時間）を計測する。③、④は、点火プラグを最も効率良く点火するためのタイミングを表した波形である。これは、点火時期を制御する信号の立ち下がりと、エンジンの回転に同期した信号の立ち下がりの間の時間を計測する。このパルスカードは、IOユニット中で使用した。

エンジンのクランク角位置を計測するセンサとしてクランク位置センサがある。このセンサはエンジンの回転に同期して信号を発生し、エンジンが一回転する角度の360度毎に信号発生するタイプや、30度毎に発生するタイプがある。ADSからも同様の信号を発生するために、IFユニット中で、IOユニットから入力した信号を分周して30度信号、180度、360度、720度等の信号を作りECUに出力している。

更に、ECU内でマイコンの動きをチェック、またモニターするために、制御に使われているRAMの計測も行っている。この計測には、IFユニット中で、専用に開発したエミュレーションカードを使用している。

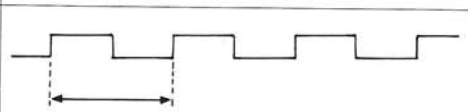
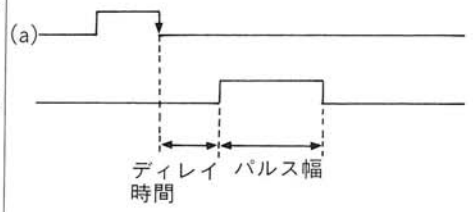
3. 1. 2　ソフト構成

ADSシステムのプログラム開発には、C言語を使った。また信号の入出力部分はアセンブラを使用した。ADSの処理は大きく分けて、設定および表示部・出力および計測部・FB（FeedBack）調整部の3つから成っている。図－3は処理の流れを示したものである。

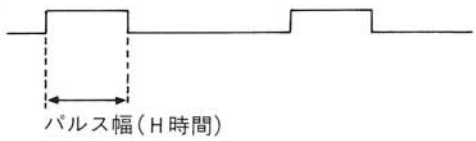
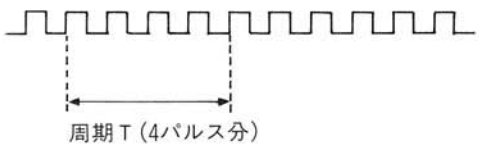
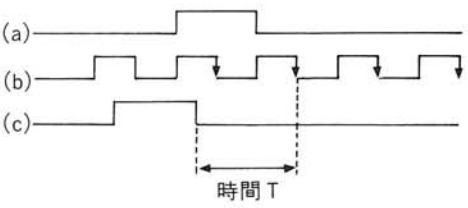
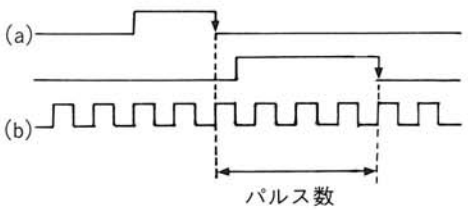
まず、設定部では、ECUに出力するセンサ信号を、仕様で要求している状態に設定するための画面を制御する。例えば、バッテリ電圧13Volt、

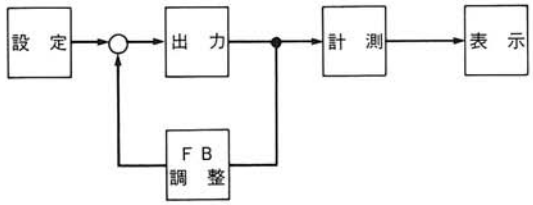
表－3　パルスカードの特性

パルス出力

No.	特　性	波　形
①	定周期のパルスを出力する ○周期は、1μs～65536μsの間で指定 ○TTLレベルで50%デューティ ○2CH指定可 （使用例）クランク角センサ信号出力	
②	信号(a)をトリガにし、指定時間遅れて、指定パルス幅の信号を出力する ○ディレイ時間、パルス幅の指定は、5μs単位 （使用例）IGフェイル信号出力	

パルス計測

No.	特　性	波　形
①	パルス幅を計測する(1パルス) ○Hの時間またはLの時間を計測する(切替可) ○3CH指定可 ○計測クロックは、1μs、5μs、100μs、1ms、又は外部クロックの切替可 （使用例）噴射量計測	
②	2パルス以上の周期を計測 ○パルスの個数は指定可 （使用例）車速計測 エンジン回転数計測	
③	2指定パルス間の時間を計測 ○(a)信号がHの時の(b)信号の立ち下がりをカウント0として(c)信号の立ち下がりからカウントが1、7、13、…までの時間を計測する （使用例）点火進角計測	
④	2指定パルス間のカウンタを計測 ○(a)信号の立ち下がりから(b)信号の立ち下がりの間のカウンタのパルス数を計測する （使用例）点火進角計測 〔基準パルスカウント〕	



図－3 処理の流れ
Fig. 3 Flow of process

吸気管内圧力100kPa、水温85℃、エンジン回転数1500rpm、アイドルスイッチONというように各信号形態に合わせた値で設定する。或いはソフトチェックを行い易いよう、CPU値でも直接設定することができる。また、設定したデータはエンジン制御プログラムの機種毎に管理している。計測部では、計測データを計測画面に表示する。また必要に応じ、指定した信号の計測値をプリントアウトする。

次に、出力および計測部では、設定部で設定した物理量のデータを、パソコンの内部データに変換してIOユニットに出力する。エンジンの状態を表すセンサ信号の種類は、様々であるため自由に対応しなければならない。そこで、変換するための関数テーブルを前述した設定部に用意して、簡単に変更が行えるようにした。また、ECUから入力する信号を計測し、パソコンの内部データから計測部で指定した物理量に変換する。

FB調整部は以下のような理由のために構成する必要があった。即ち、自動車制御機器のように、アナログ、パルス、デジタル等多様な形態の信号を処理し、その結果複数の出力信号を微妙に制御する装置においては、入力される信号の僅かな誤差によりプログラム処理の流れが大きく変わることもある。そのため、ECUへの入力信号の設定は正確に行った上で、デバッグを行う必要がある。ところが、出力値をいくら正確に設定してもECU内の処理回路の誤差（AD変換による誤差等）により、マイコンには設定した値が誤差を含んだ値として入力される事になり、正確なデバッグを行う事ができない場合がある。このため、ECUへの入力値ではなく、ECUに使われているマイコンが読み取った値（例えばAD変換結果）と設定値が一致するように調整する必要がある。

そこで、ADSではFB調整を行うことによって設定値通りの値を出力するようにしている。図－4は、ADSにおけるFB調整の原理を表した図である。

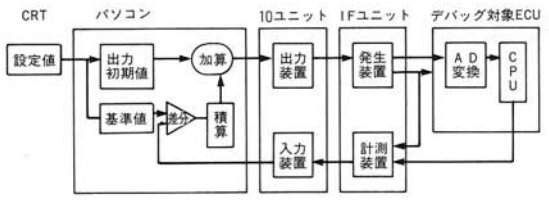
ECUに出力する信号のFB調整を、ECUのコネクタ端子の値で行った場合、ハードの誤差も含めたチェックを行うことになる。これに対し、ECU内のCPUが読み取った値（RAM値）でFB調整を行った場合、制御プログラムのチェックを行うことになる。ADSでは、目的に応じ選択することができる。

3. 2 機 能

主な機能として次の4つをあげる。

① 設定・計測

各擬似エンジン信号の設定とECUの出力信号の計測を行う。



図－4 F B調整
Fig. 4 Feedback adjustment

② 連続計測

信号の設定と計測を連続的に自動で行い、結果をプリントアウトする。

③ デバッグパターンの自動生成

エンジン制御プログラムに対応して、デバッグのパターンを自動的に作成する。

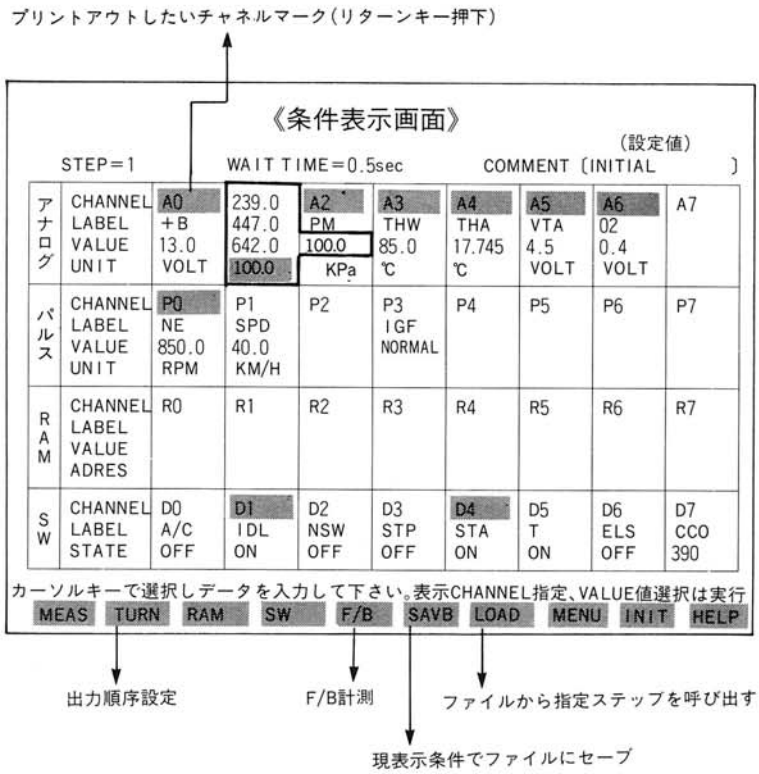
④ スウィープ計測

ECUへの出力値を連続的に変化させた時、ECUの出力を計測する。

3. 2. 1 設定および計測

デバッグを行うためにはセンサ信号の値を設定して、あるエンジンの状態を作りだす必要がある。各センサの設定は、パソコンの画面上で行う。この機能では、設定した値をECU内に出力し、ECUの制御出力の計測値を見て、ECUの出力値が仕様を満たしているかどうかの確認を行う。エンジン条件の設定は、キーボードから物理量で行うので、簡単に変更することができる。設定および計測条件は自動計測の1ステップとして登録する。登録の順番は任意であり、追加・削除も自由である。また、どのステップでも何度でも変更可能である。この機能を有効に活用するためには、使い易いMMI（Man Machine Interface）になるよう、設定画面と計測画面を作成する必要がある。

例えば、設定画面は図－5の通りである。設定あるいは計測の全チャネルは1画面上で表示するようにし、また設定画面と計測画面は同形式で、ワンタッチで切替えるように設計した。設定条件は、アナログ、パルスの各センサ信号の場合は物理量またはCPU値を、デジタル信号の場合はON・



図－5　設定画面

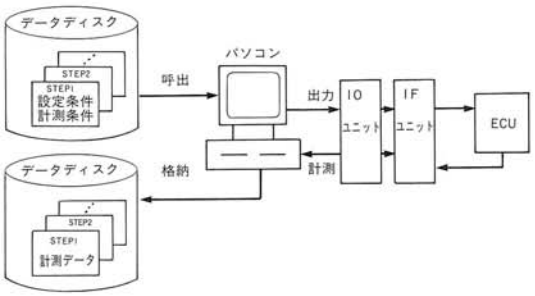
Fig. 5　Setup display

OFFの状態を入力する。計測結果をプリントアウトしたいチャネルは、CHANNEL部分にカーソルを合わせ実行キーを押下し、マークすることで選択する。また、ファンクションキーに次のような機能を割り付けた。例えばTURN機能では各信号をECUに出力する順番の設定を行う。これは、設定した信号は通常アナログ信号チャネル0から順次ECUに出力するが、エンジン制御プログラムによっては出力する信号の順番によって制御内容が変わることもあるため、出力順序設定の機能を付加した。FB機能では現在、表示設定している値になるようFB調整を行う。SAVE機能では、現在表示している設定値を自動計測用の条件設定ファイルにセーブする。LOAD機能では指定ステップの設定値を自動計測用の条件設定ファイルからロードして画面上に表示する。

3. 2. 2 連続計測

連続計測の概念を図－6に記す。

デバッグを行う際は数千項目の条件設定および計測をする必要がある。その1つ1つの設定および計測を1ステップとし、デバッグ項目毎に設定条件及び計測項目をファイルに登録し、それらを順次呼び出すことによって、連続的に計測を行うことにより、数千ステップに及ぶデバッグの自動化が可能となる。



図－6 連続計測
Fig. 6 Sequential measurement

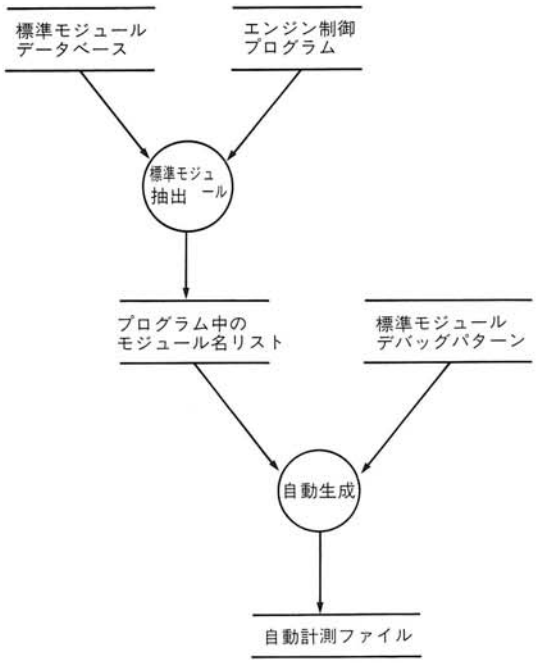
ファイルに登録する事によって、1度設定すれば同一条件での計測を繰り返し行えるので、既に計測したステップを確認のためもう一度計測したい場合や、一部を変更して他の機種に流用したい場合は、登録したファイルを呼び出し再計測を行う。仕様変更等でエンジン制御プログラムの内容を変更をした場合、たとえ部分的な変更であっても、デバッグは、変更箇所だけではなく、プログラム全体に影響がないかどうかをみるために、もう一度全体のデバッグを行う必要がある。従って、僅かな変更のために多くの工数をさかなければならなかった。ADSでは、設定条件および計測項目を自動計測用条件設定ファイルに登録しているので、ファイルを部分修正したあと再度自動計測を行うだけでデバッグに対応する事ができる。従って、仕様変更の場合は従来に比べ格段にデバッグ工数を削減することができる。

3. 2. 3 デバッグパターンの自動生成

デバッグパターンの自動生成の概念を図－7に記す。

デバッグ作業は前述した通り、仕様書に記述されている処理の実行を一つ一つ確認することで行われる。従って、仕様に応じてECUに入力するセンサ信号の条件を設定する必要があるため、デバッグパターンは1ステップづつ地道に作成していかなければならないが、これでは作業効率の向上は図れない。

ところで、自動車のエンジン制御プログラムは、設計効率を上げるためモジュールプログラミング方式で設計を行っている。従って、各機種で共通なモジュル（標準モジュール）は、デバッグ項目も一定しているので、各標準モジュールに対応したデバッグパターンを予め作成しておけば、そのパターンを組み合わせることによりデバッグ条件



図－7　デバッグパターンの自動生成
Fig. 7　Automatic generation of debug pattern

を登録した自動計測用ファイルは簡単に作成することができる。

ADSでは、既に登録している標準モジュールのデータベースを参照し、エンジン制御プログラム中で使われているモジュール名を抽出して、それに対応したデバッグパターンを自動的に自動計測用条件設定ファイルに配置している。これにより、自動計測用ファイルを作成する効率を向上することができる。また、RAM計測を行う際アドレスを指定する必要があるが、エンジン制御のプログラムの機種によりアドレスに相違がある。このため、RAMのラベルをもとにエンジン制御プログラムのリストを検索し、自動的にアドレス設定をするように工夫している。

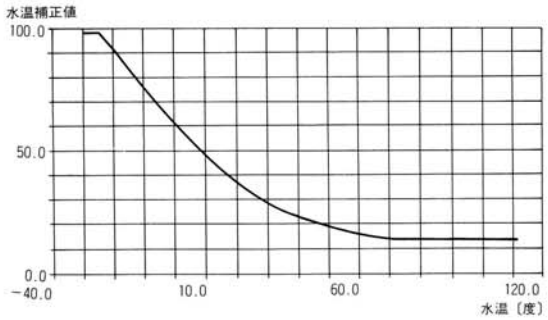
標準化の進んだエンジン制御プログラムは、殆ど自動的にデバッグパターンを作成することができる。

3. 2. 4　スウィープ計測

今まで、設定条件を設定してそのポイントで計測を行うという、定常的な計測方法について述べてきたが、この方法は、出力値を下限から上限まで変化させて（スウィープ）計測を行う。

例えば、水温センサの信号を－30℃～＋110℃の範囲で連続的にスウィープさせた時、ECU内のマイコンが制御する水温補正値は、図－8のように連続的な曲線を描く。エンジン制御では、気温や水温等の環境に応じ制御量の補正を行っている。この補正は図－7中の表のように、各温度に対応した補正量を持っおり、温度間の補正は補間計算によってもとめる。デバッグをする時は、補正量が全温度範囲で正常に計算されていることを確認する必要がある。ADSでは出力値を下限から上限まで変化させ、その間の補正量を計測することにより、全範囲における補正のチェックを行う。また、グラフ表示することにより、補正値に異常の無いことが一見してわかる。つまり、水温補正値は要求仕様通りの動作をしていることが確認された訳である。

通常のデバッグによって、制御出力の全点チェックを行おうと思ったら、大変な時間と労力をかけなければならない。また、多数のためチェック洩れを生じる可能性もある。このため実際のチェックは、代表的な点を絞り、その部分で確認を行っている。ADSでは、自動化する事により、全点チェックが可能となり信頼性の向上を図ることができ、作業効率を向上させることもできる。また、通常のデバッグの場合、特に注意しなければならないのは、仕様で指定した範囲外でも、本当に一定値をとるかどうかである。通常のデバッグでは、範囲外の、代表的な1点のみのチェックとなっている。ADSでは、スウィープする信号の範囲は



(水温－水温補正値)

水温	度	−24.75	−7.920	4.7200	16.534	29.236	45.054	70.182
	CPU	1C	3C	5C	7C	9C	BC	DC
水温補正値		98.30	72.70	55.295	40.448	29.184	20.992	13.824
	CPU	C000	8E00	6C00	4F00	3900	2900	1B00

図－8　スウィープ計測出力例（水温－水温補正値）
Fig. 8　Output sample of sweep measurement

任意に決定することができるので、出力値の下限を仕様にある温度範囲よりも低く、また上限を高く設定しておけば、指定範囲外の動作チェックを行うことも簡単である。

水温と水温補正値の任意の格子点について、計測値を表にして出力している。また、計測値を指定単位系に変換する事も行っている。更に、物理量とRAM値の両方を表示することにより、プログラムのチェックを容易にしている。

4. あ と が き

ADSにより、デハッグ作業の自動化を図る事で、人為的ミスをなくし、信頼性を向上させる事ができた。さらに、作業効率の向上にも役立っている。定常的なデハッグにおいては20%程度の時間短縮に止まるが、ある程度仕様の固まったものはデバッグパターンを登録していけるので、仕様変更や定数変更の場合のデバッグは60%程度の時間短縮をすることができる。基本的には、定常的なデバッグはADSで置き換えることができ、またエンジン制御プログラムは製品化されるまでに何回となく試作を繰り返すため仕様変更や定数変更のデバッグが多くなるので、ADSを使用することによりプログラム開発全体では50%程度の工数を削減することができる。また現在、制御の変化点を自動的に検索するような過渡的チェックを行えるように機能を追加しており、この機能ができれば更に効率向上につながると思われる。

ADSでは多数のパターンを計測しているため、仕様通りの制御をしているかどうかを計測結果から確認するのに時間がかかる。この対策として、自動判定機能を計画中である。これは、仕様変更以前の計測データと今回計測データとの比較、或いは理論上求められる制御量と計測値との比較を自動的に行い、比較結果を一目で判るように出力することである。このようにADSはデバッグ作業をプログラム化することが可能なシステムである。多様化するエンジン制御機器のプログラム評価にも柔軟に対応できるものであり、ADS使用を前提としたデバッグ項目の設定や作業のプログラム化等利用面を中心としたデバッグ工程の見直しにより、更に作業時間短縮・信頼性の向上が図れるものと考える。