3M-10

組込み型ソフトウエア開発環境
－その1　システム概要－

菊地一成，岩渕洋一，渡辺浩康，浅野俊昭
キヤノン㈱　情報システム研究所

1.　はじめに

近年では、データフローに基づく機能分解法による生産性への寄与が注目されつつある。しかしながら、複写機やＬＢＰの制御といった並列性の強い機器の組込み型制御ソフトウエアにおいては、依然有効な開発技法がないのが現状である。このため、我々はこの領域におけるソフトウエア開発を支援するシステムの開発に取り組んでいる。本報では、現在開発中のシステムの概要を報告する。

2.　研究の課題

本領域のソフトウエア開発の問題として、

①仕様書は、時間や同期関係を表現する適切な記述法がなく、完全性、伝達性が悪い。

②プログラムは、上記同期関係が非明示的な形式で表現されるため、組込み機器の動作との対応関係が悪い。このため、生産性、保守性が悪く、プログラム検証にも問題がある。

③テストは、ＩＣＥ等による実機テストが主体で効率が悪く、また設計の上流工程で仕様の検証が困難である。と言った問題がある。

これらの問題を解決するためには、設計対象機器と対応関係の良い並列制御の仕様表現法及び、それからのシステム動作の検証法とプログラム生成法の開発が課題となっている。

3.　システムの概要

本研究は、これらの問題を解決するため、ＳＲグラフと呼ぶプロセス間の同期関係を明示的に定義しうる仕様表現法に基づき、同期関係に基づくプログラム骨格の自動生成、自動検証を行うシステムの構築を目指している。

本システムでは、まずプログラマはグラフィックエディタを用い後述のＳＲグラフの表記法に従う仕様を入力することになる。入力された部分は、逐次動的な状態のシミュレーションが行われる。設計者は、この結果を見て、機器の動作が意図したように定義されていることを確認しながら入力を続け、最終的には正しいＳＲグラフから自動的に機器を制御するプログラムコードを得ることが出来る。

4.　ＳＲグラフ

組込み機器内の各ユニット固有の制御シーケンスに関する記述と、それらのユニットが機器として協調動作するための同期条件に関する記述が分離されているほうが、仕様記述と対象機器間の対応関係が良くなり設計者の理解し易い記述法となる。また設計の途上において、設計者の集中力が損なわれないためには、これらの記述が１つの記述形式の中に統合されている必要がある。そこで、以上の条件を満たす仕様記述法として、図１の仕様記述例で示されるＳＲグラフを考案した。以下に、このＳＲグラフの仕様を示す。

ＳＲグラフは次の四つの基本要素から成る。

①　ＰＡ（Primitive Action）

システム内の各ユニットがある時点すべき詳細手続きであり、図中○で記される。ＰＡはその手続きの実行中であることを他のユニットに知らせるためのフラグ（後述）を出す機能を有する。

②　遷移

ＰＡの実行順序のつながりを示すもので、図中↓で記される。↓で接続された一連のＰＡ群は各ユニットの制御シーケンスを構成する。遷移はそれが発生したことを他のユニットに知らせるためのトリガ（後述）を出す機能を有する。

CASE System for an embedded Software
Kazushige KIKUCHI, Youichi IWABUCHI,
Hiroyasu WATANABE, Toshiaki ASANO
Canon inc.

③　フラグ

制御シーケンス間の同期関係を記述するためのもので、図中⇢で記される。フラグは、あるＰＡ実行中に、他のユニット内の特定の遷移に対し遷移の許可を与えるのに用いる。

④　トリガ

制御シーケンス間の同期関係を記述するためのもので、図中⇢で記される。トリガは、ある遷移の発生と同期して他ユニット内の特定の遷移を強制的に引き起こすのに用いる。

なお、以上の要素により、AllenとHayesの時区間論理［１］で述べられた２つの時区間に存在する１３個の基本的関係の全てが表現可能である。

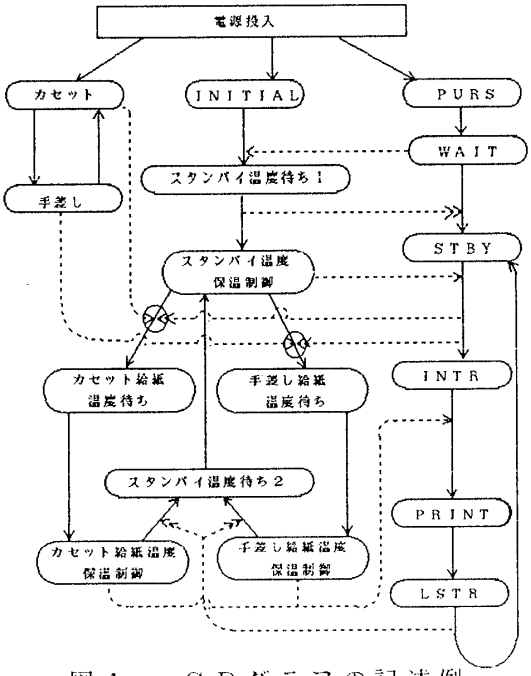

図１．　ＳＲグラフの記述例

５．　ＳＲグラフの検証

機器に組込まれるタイプのソフトウエアでは、製品流通後における機能の変更が困難なため、特に高い信頼性が要求されている。しかしながら、現状のプログラムにおいては、各ユニット制御間の同期関係が、共有メモリの利用等といった形式で非明示的に埋め込まれてしまうため、機器の動作の把握を難しくしている。このため、バグの検出、その除去、テストに多大な労力を必要とする原因となっている。ＳＲグラフでは、この同期関係の記述を明示的な記述にし、更にＰＡ内の各ユニットの詳細手続きとは分けて取り扱える様に形式化した。こうして、ソフトウエア開発の早い時期で、同期関係に基づく機器の動作シミュレーションの実施が可能となった。この検証は、ＳＲグラフの合成と呼び、図２に示す様にＳＲグラフから同期関係の記述を消去することによって、実際の機器の動作の状態図を得ることで行われる。

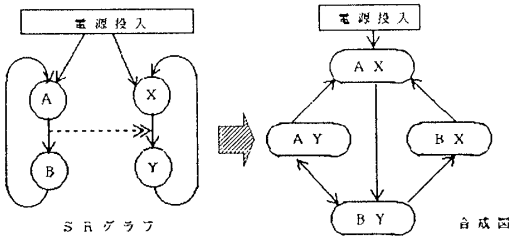

図２．　ＳＲグラフの合成

６．　プログラムの自動生成

ＳＲグラフでは、ＰＡ内の詳細な制御手続きと、各ユニットのＰＡの実行順序及びそれら間の同期関係が独立して定義される。そのためMannaとWolperの発想［２］と同様に、システムとしての協調的動作を与える同期関係の記述部分に基づき、プログラムの骨組みを自動生成することが可能である。更に、この骨組みに各ユニット制御の詳細手続き（ＰＡの内容）をユーザが書き込むことによって最終的なプログラムが得られる。

７．　おわりに

現在までに、実際の製品の一部仕様のＳＲグラフから仕様の検証とＣ言語のプログラム生成の機能実験を終え、その有効性を確認した。今後は、時間的制約を考慮したより厳密な検証法と、効率良いプログラムコードの生成法を検討する予定である。

参考文献

［１］Allen and Hayes, "A Common-Sence Theory of Time", IJCAI-85, 528-531

［２］Manna and Wolper, "Synthesis of communicating processes from temporal logic specification", ACM Trans. on Lang. and sys., vol.6