状態遷移表を中心としたマイコン組み込みシステム用開発環境 ZIPC ～ 具体的応用例 ～

7D－8

矢野 渡†、渡辺 政彦‡、奥村 晃子＊
†NEC マイコンテクノロジー (株)、‡テスコ (株)、＊NEC(株)

1 はじめに

ZIPC V4.0w(以下 ZIPC) は状態遷移表を中心とし、上流の設計工程から下流のデバッグ工程までをシームレスに接続した、マイコン組み込みシステム用ソフトウェアを開発するための CASE ツールであり、設計レベルでの動的検証、ソースコードの自動生成、設計書を生かしたオブジェクトのデバッグなどといった機能を提供している。

従来、マイコン組み込みシステム用の CASE ツールは市場にほとんど無かった。その理由はいくつか考えられるが、実際の製品開発に活用できるだけの実用性がどこまで確保できるかが最も重要な問題である。

本稿では、ZIPC を用いて実際にアプリケーションシステムを開発し、以下の諸点から、開発環境としての ZIPC の実用性を検証した結果を報告する。

(1) 設計レベルでどこまで検証作業が可能か

(2) 自動生成したオブジェクト効率はどうか

(3) デバッグ工程をどれだけ支援できるか

2 開発対象と環境

2.1 開発対象システム

ケーススタディとして、アナログコードレス電話デモセット (NEC 製 μPD78078 使用) 用ソフトウェアの再開発を行った。

図 1 に開発時の機器接続イメージを示す。

2.2 開発環境

デバイスに特化した部分の開発を行うため、ZIPC

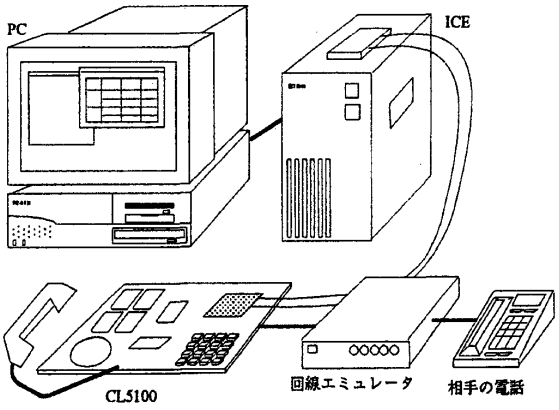

図 1: CL5100 機器接続イメージ

を NEC 製マイコン開発ツールと接続した環境で作業を行った。

図 2 にツール類の接続関係を示す。

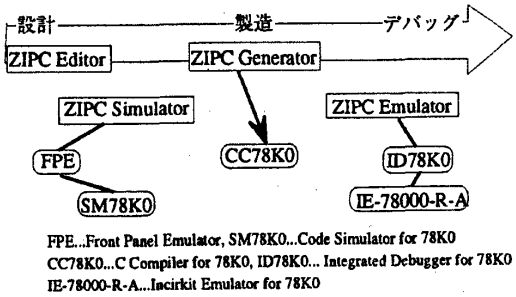

図 2: ZIPC とマイコン開発ツールとの接続

3 開発の流れ

本章では実施した開発作業の流れについて示す。

3.1 流用部分の切り分け

今回の開発作業では、ハードウェアに密着した部分と、時間的制約が厳しい部分は既存のアセンブラコードを流用した。(システム全体のほぼ半分)

3.2 設計

(1) 電話機の仕様から、システムの状態と発生し得る事象とを抽出し、状態遷移表を作成。

"ZIPC: The STM Based Software Development Environment for Embedded Microcomputer Systems. – Examination", Wataru YANO†, Masahiko Watanabe‡and Akiko Okumura*, †NEC Microcomputer Technology, Ltd., ‡TESCO Ltd., * NEC Corporation

(2) 電話機の外観イメージをCRT上に表現したFPEと接続した設計仕様の動的な検証。...(図3)

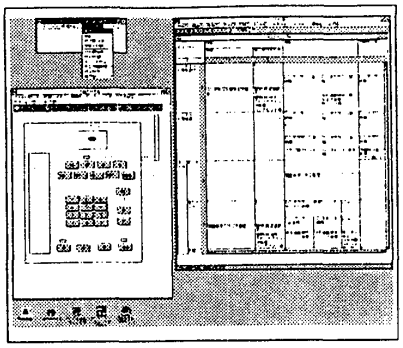

図 3: 設計段階での動的検証

3.3 製造

(1) 作成され、設計レベルでの検証が済んだ状態遷移表から、Cソースコードを自動生成。

(2) Cコンパイラ(CC78K0)でコンパイル。アセンブラ流用部とリンクし、オブジェクトを作成。

3.4 デバッグ

ターゲットボード上のオブジェクトを、状態遷移表、ソースコードの双方でデバッグを実施。

作業中の画面イメージは図4を参照。

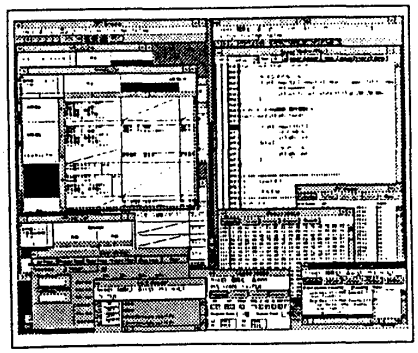

図 4: 設計書でのコードデバッグ

4 開発結果とツールの効果

4.1 設計レベルでの検証

今回の開発作業では、以下のような設計レベルでの検証を行うことができた。

(1) 状態遷移表を利用することにより、想定すべき事態の洩れなどが無くなった。

(2) FPEと接続した動的検証を実施したことにより、設計レベルでシステムの論理的動作をすべて確認することができた。

このように設計レベルでの検証を実施できたため、仕様ミスによるバグ、後戻り作業はなくすことができた。結果的に工数の削減が可能になった。

4.2 オブジェクト効率

今回の作業で、状態遷移表から自動生成したCソースコードによるオブジェクトの効率は、既存の全アセンブラ記述によるオブジェクトとほぼ同等の効率が得られた。両者を比較した結果を表1に示す。

表 1: オブジェクトサイズの比較

単位: バイト

	ROM			RAM		
	ユーザ	OS	計	ユーザ	OS	計
従来	22080	2795	24875	303	430	733
今回	11803			303		
	9982	2512	24297	20	438	761

このような結果が得られた原因の一つとして、フラグ判別のための条件分岐コードが、テーブル駆動型のプログラム構造になったことで、大きく削減されたことがあげられる。

4.3 状態遷移表によるデバッグ

ターゲットボード上でのオブジェクトの動作を状態遷移表の上で見ることができ、またブレークポイントの設定などのデバッグ作業を行うことができたため、以下のような利点が有った。

- バグ発生時にも、システムの状態と事象が明確なため、原因追求が以前より容易。
- 設計書イメージのままデバッグ作業が実施できたので、仕様との関連付けが明確。

これらの利点により、以前よりも快適なデバッグ作業を実施することができた。

5 おわりに

今回の作業でZIPCがマイコン組み込みシステム用開発環境として実用的なことが実証できた。今後、適用可能なシステム領域の拡大など、より一層の充実を図って行きたい。