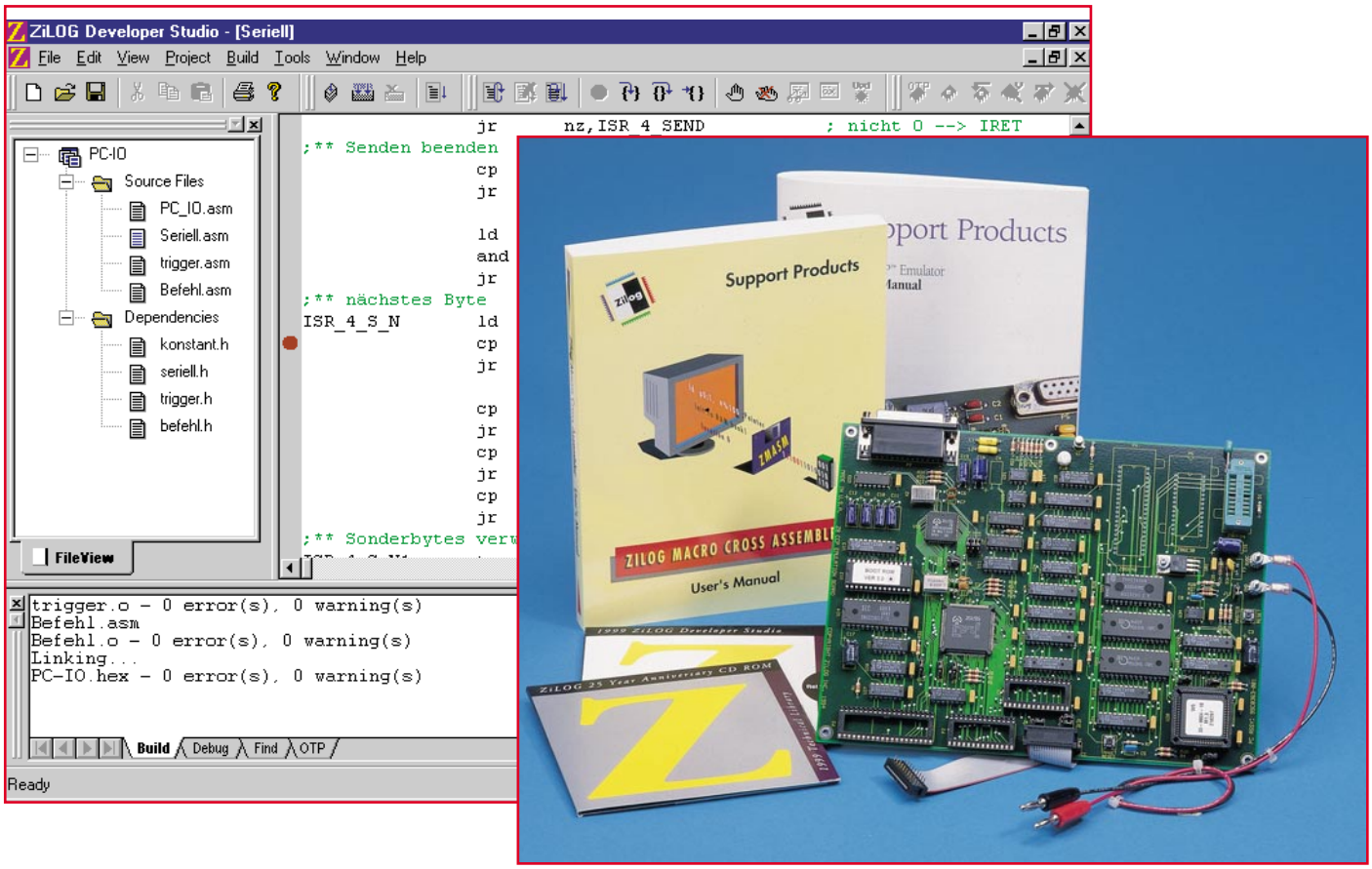

Z8-Entwicklungstools

Nach der Vorstellung des Z8-Emulatorboards im „ELVjournal" 3/99 geben wir eine Übersicht über die umfangreiche und komfortable Entwicklungsumgebung für diese Mikrocontroller-Reihe.

Das Developer Studio

Hat man von der dem Emulator-Kit beiliegenden CD-ROM „Development Software" (für Windows 95/98/NT) aus das komplette Setup ausführen lassen, finden sich in der Programmgruppe „ZILOG" das ZILOG Developer Studio (ZDS), das DSPICE- und das ZPROG-Tool, wie bereits im „ELVjournal" 3/99" beschrieben.

Das GUI ist von einer getrennt beiliegenden Diskette zu installieren, ebenso bei Bedarf der Assembler ZMASM.

Schwerpunkt unserer Softwarevorstellung soll das ZDS bilden, da dieses wohl auch das meistbenutzte Tool des Softwarepaketes sein wird.

Vor allem aus Platzgründen beschränken wir uns jedoch auf die globale Beschreibung des Funktionsumfangs, die „Bedienungsanleitung" des ZDS allein würde den Rahmen dieses Artikels sprengen, zumal die Programmierarbeit denn doch schon

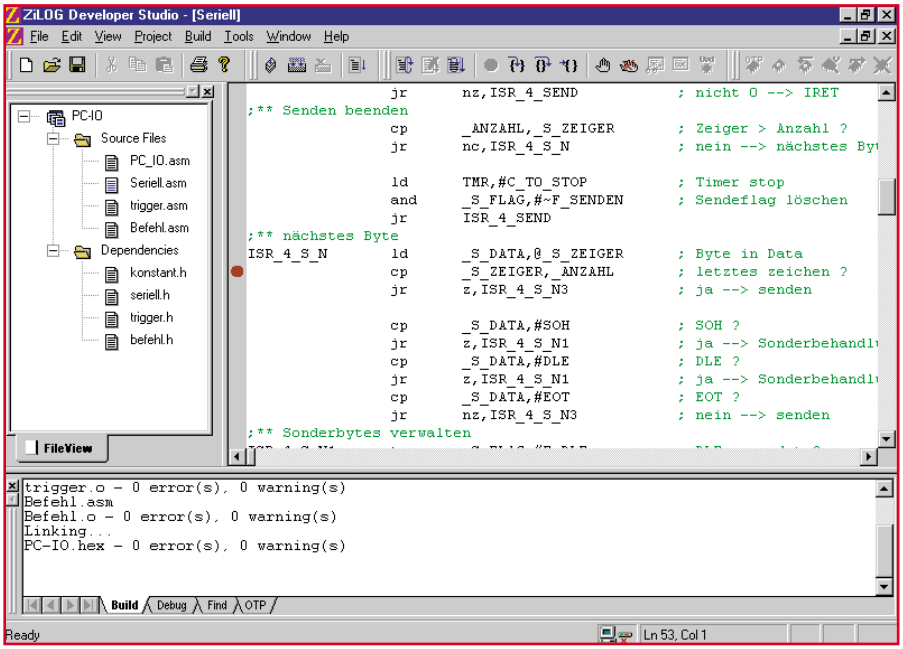

Bild 1: So präsentiert sich der ZDS mit seiner komfortablen Oberfläche

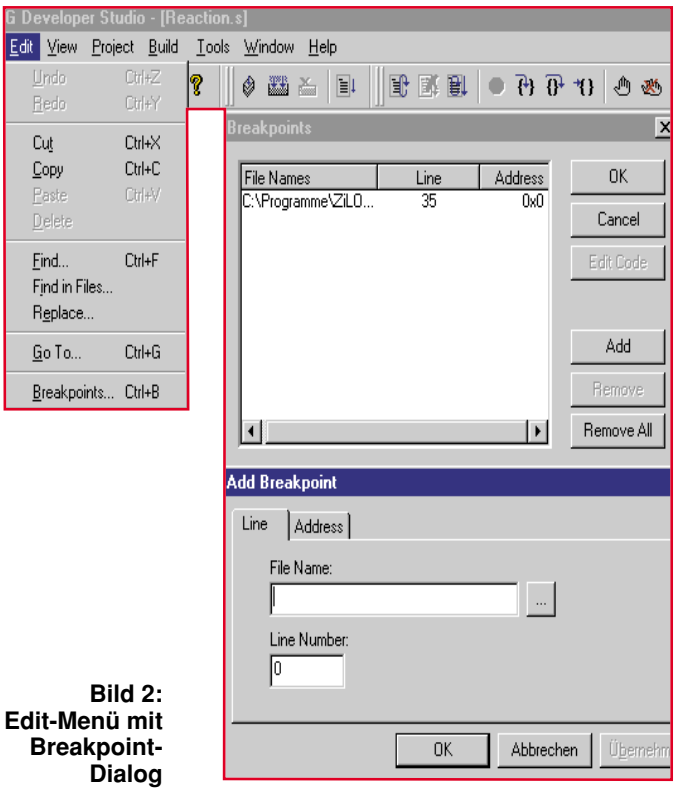

Bild 2: Edit-Menü mit Breakpoint-Dialog

mehr als nur einige Grundkenntnisse voraussetzt.

Startet man das ZDS, erscheint in zunächst 3 Fenstern eine äußerst komfortable Oberfläche, die starke Ähnlichkeiten zu Microsofts „Visual Studio" aufweist. Programmierer werden dieses Erscheinungsbild begrüßen, wird ihnen doch die Umstellung nicht schwerfallen.

Das linke der drei Hauptfenster verwaltet die Projektliste.

Hier sind übersichtlich die eigentlichen Projektdateien und die sogenannten Dependencies (Abhängigkeiten) des aktuellen Projekts aufgelistet, so daß man besonders bei mehreren Teilprojekten und längeren Programmen mit vielen Dependencies eigentlich kaum einmal die Übersicht verlieren kann.

Das rechte (größte) Fenster ist das Editorfenster, hier erscheint das Quellfile.

Schließlich finden wir unten das Resultfenster, in dem die Auswertungen z. B. von Compiler-Läufen, Debugger-Läufen usw. erscheinen.

Abbildung 1 zeigt die gesamte Bedienoberfläche mit einem Beispielprojekt.

Windows-üblich erreicht man über die Menüzeile alle Menüs und dazu direkt über die darunterliegende Symbolzeile zahlreiche oft benötigte Funktionen, was die flinke Arbeit mit dem Programm sehr erleichtert.

Mit dem Datei-Fenster (File) eröffnen sich die üblichen Möglichkeiten zum Öffnen, Speichern, Neuerstellen, Drucken und Verwalten von Projekten.

Über das Edit-Menü sind sowohl allgemein übliche Editiertätigkeiten am Quelltext wie Undo, Cut, Copy, Paste usw. als auch Such- und Ersetzungsfunktionen, Sprünge zu einer bestimmten Programmzeile und schließlich das Setzen von Breakpoints über einen mehrstufigen Dialog möglich.

In Abbildung 2 wird u. a. dieser Dialogweg angezeigt.

Wer das „MS Visual Studio" bereits kennt, wird hier wohl sofort versuchen, den Breakpoint auf die dort übliche Weise in die Laufleiste links des Editorfensters zu setzen, was sich auch als möglich erweist (Abbildung 3).

```
;** nächstes Byte
ISR_4_S_N       ld      _S_DATA,@_S_ZEIGER
                cp      _S_ZEIGER,_ANZAHL
                jr      z,ISR_4_S_N3
```

Bild 3: Breakpoints sind auch bequem direkt setzbar

Da erscheint auch der „Set Breakpoint"-Button in der Symbolleiste fast schon überflüssig.

Das View-Menü erlaubt die Gestaltung der Arbeitsoberfläche, also die Auswahl, welche Fenster aktuell geöffnet sein sollen.

Einzige Besonderheit ist hier die Refresh-Möglichkeit, um z. B. die Anzeige im Projektfenster nach Änderungen, etwa dem Erstellen neuer Dependencies, zu aktualisieren.

Der nächste Menüpunkt, das Projekt-Menü, macht zunächst die individuelle Einstellung von Assembler, Linker, Editor, Debugger und OTP-Link möglich.

Des weiteren wird hier die Möglichkeit geboten, Projektteile zu ergänzen, also nachzuladen, oder auf einfache Weise zu entfernen.

Schließlich erfolgt hier über „Target" die Auswahl des zu „behandelnden" Prozessors und des angeschlossenen Emulatorboards aus einer umfangreichen Bibliothek, wahlweise geordnet nach Applikationen oder Prozessorfamilien.

Daneben sind auch die Daten des gewählten Prozessortyps aufrufbar, so daß man sich Nachschlagen nach Daten weitgehend ersparen kann.

Abbildung 4 vermittelt einen Eindruck vom Angebot der Database.

Die Option „Emulator Configuration" erlaubt die Einstellung der Betriebsart des angeschlossenen Boards (bei Zilog „ICE-BOX" genannt), die Auswahl der seriellen Schnittstelle und der Datenübertragungsrate.

Im Menü „Build" findet man alle zum Assemblieren/Compilieren, Aktualisieren und Debuggen wichtigen Funktionsaufrufe (Abbildung 5).

Wichtig ist hier auch der Connect-Aufruf, also das Herstellen der Verbindung

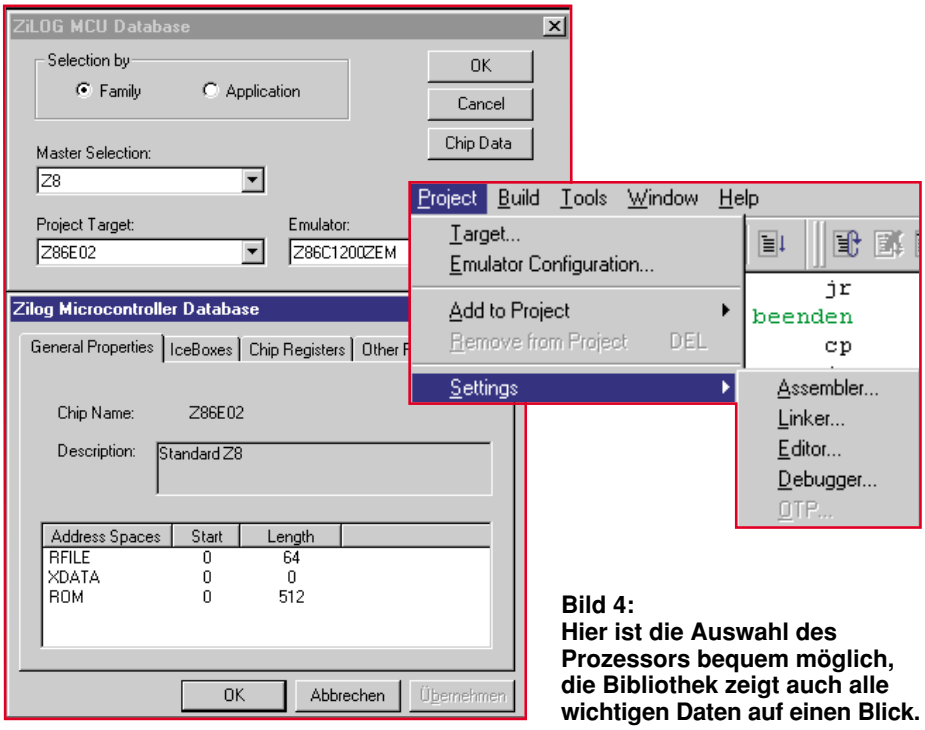

Bild 4: Hier ist die Auswahl des Prozessors bequem möglich, die Bibliothek zeigt auch alle wichtigen Daten auf einen Blick.

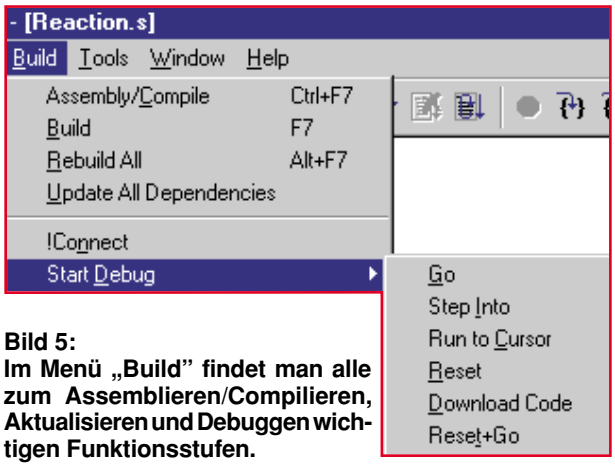

Bild 5:
Im Menü „Build" findet man alle zum Assemblieren/Compilieren, Aktualisieren und Debuggen wichtigen Funktionsstufen.

zum Emulator-Board nach vorheriger Konfiguration.

Ist die Verbindung zum Emulator-Board hergestellt, kann der Code über die Option „Start Debug" von hier aus in den Emulator heruntergeladen werden.

Schließlich bietet das Menü „Tools" die Möglichkeit der Einstellungen für die OTP-Programmierung, eines Firmware-Upgrades des Emulator-Boards und der individuellen Konfiguration der Toolbar.

Bleiben noch die Menüs „Window" und „Help", die im ersten Fall die Windows-üblichen Anordnungen aller geöffneten Fenster (Projekte) und deren Verwaltung und zum zweiten die sehr umfangreiche Online-Hilfe zum gesamten Programmsystem einschließlich Hilfe für den Kontakt zu Zilog anbieten.

Die Anleitung zum Makro-Assembler selbst findet man im übrigen auch als gedrucktes, ausführliches Handbuch im Emulator-Kit vor.

Abschließend zur Erläuterung der Bedienoberfläche des ZDS sei nochmals auf die komfortable Möglichkeit hingewiesen, über die Toolbar zahlreiche Befehle, z. B. zum Debuggen, direkt über die entsprechenden Buttons zu erreichen.

Auch der Service des Ergebnisfensters, hier alle Ergebnisse der verschiedenen Test- und Übersetzungsläufe gewissermaßen zwischenspeichern und nacheinander abrufen zu können, soll nicht unerwähnt bleiben.

Damit ist die Beschreibung der komfortablen Oberfläche des ZDS bereits abgeschlossen.

GUI und Z 8 ICEBOX

Hat man sich mit dem ZDS schon einmal beschäftigt, wird man schnell feststellen, daß man des GUIs für den Kontakt zwischen dem eigentlichen Makroassembler ZMASM und dem Emulatorboard kaum bedarf.

GUI und ICEBOX bilden zwar zusammen mit dem ZDS eine Software-Einheit, können jedoch auch separat betrieben werden. Sie sind quasi der „Vorläufer" des ZDS und je nach Vorliebe des Benutzers statt diesem einsetzbar.

ICEBOX (Abbildung 6) ist ein Debug-Werkzeug, das es erlaubt, maschinennah mit dem Prozessor zu kommunizieren.

Auch hier ist es möglich, Assembler-Files zu laden, zu editieren, zu speichern und zum Emulatorboard herunterzuladen, den Status des Prozessors (Register, Flags etc.) zu ermitteln und zu verändern, Assemblerfiles zu debuggen usw.

Im Konfigurationsfenster des GUIs (Abbildung 7) kann man ähnlich wie im „Target-Menü" den Prozessortyp konfigurieren.

Hier hat man, das ist der entscheidende Vorteil dieses Werkzeugs, die Möglichkeit, auch Prozessoren zu konfigurieren, die sich eventuell noch nicht in der mitgelieferten Konfigurationsbibliothek befinden.

Meist wird man jedoch zum „moderneren" Instrument, dem ZDS greifen, da dieser doch mehr und komfortablere Möglichkeiten für das Assemblieren, Compilieren und Debuggen bietet.

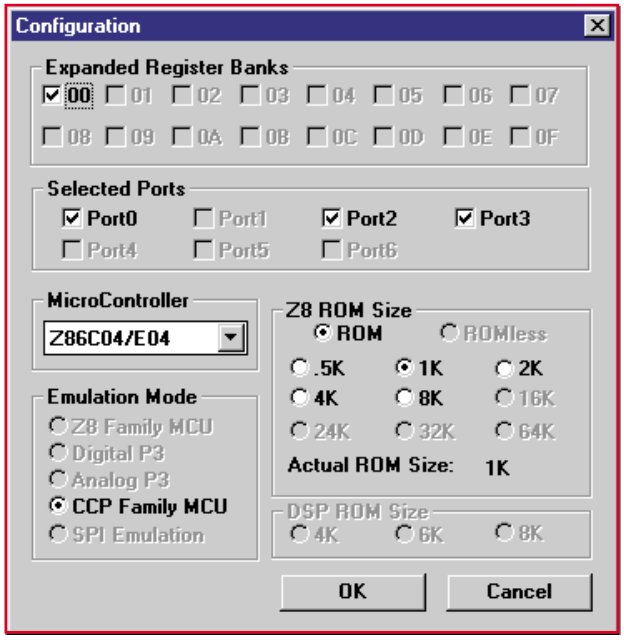

Bild 7: Die Prozessor-Konfigurationsbox des GUI

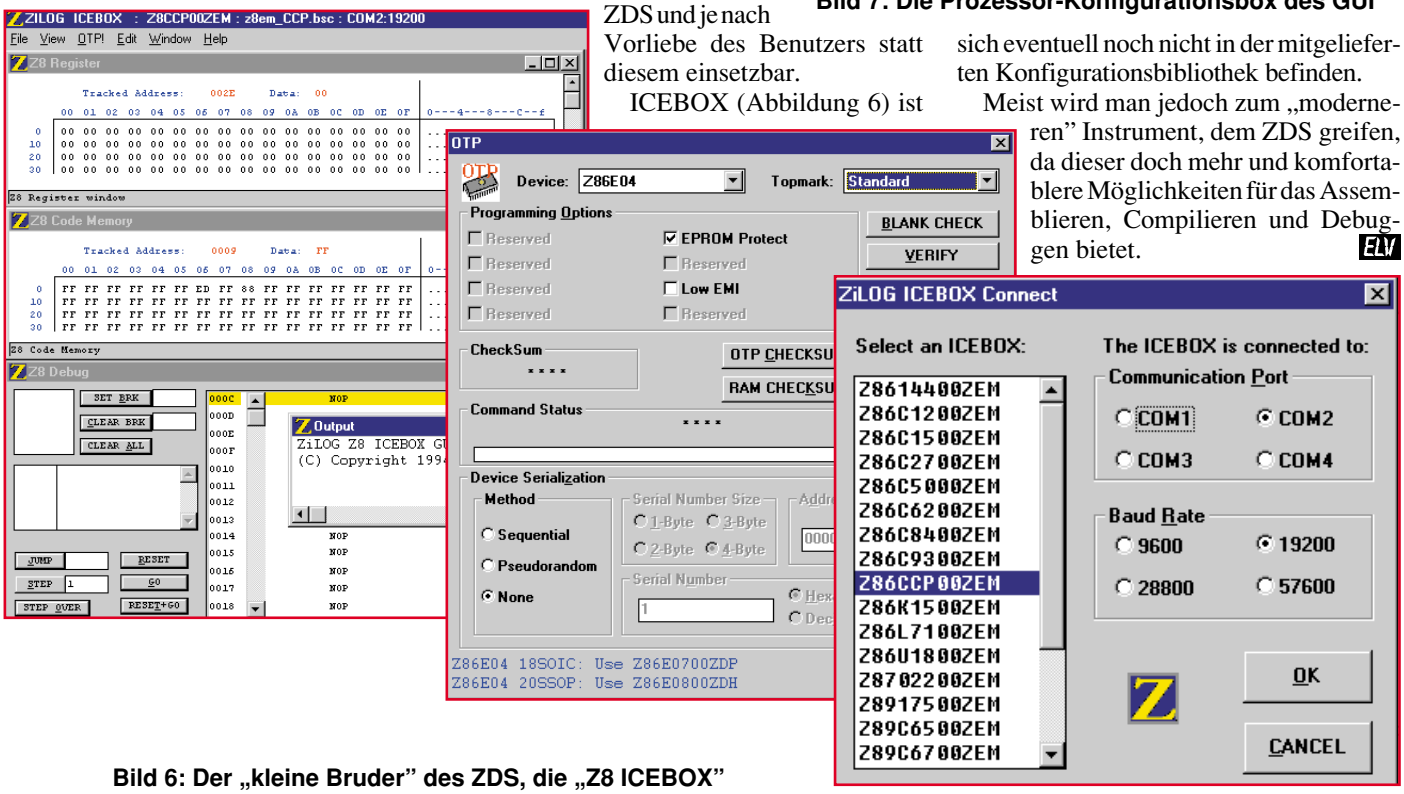

Bild 6: Der „kleine Bruder" des ZDS, die „Z8 ICEBOX"