IAR Embedded Workbench for ZiLOG's eZ80™
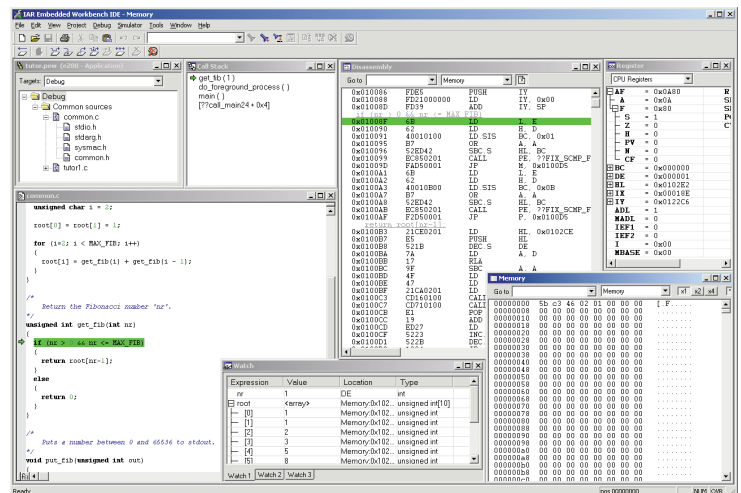
IAR Embedded Workbench IDE

IAR Embedded Workbench is a set of highly evolved development tools for programming embedded applications. It fully integrates the optimizing IAR C/Embedded C++ Compiler, assembler, linker, librarian, text editor, project manager, and C-SPY debugger in one Integrated Development Environment (IDE). The IAR Embedded Workbench for ZiLOG's eZ80® generates very efficient, reliable, and PROMable code for the eZ80® family.

Features

- Fully integrated development environment under Windows®/98/ME/NT/2000/XP
- ISO/ANSI standard C and Embedded C++ Compiler
- Built-in advanced chip-specific optimizer
- Chip-specific C extensions
- Easy and fast interrupt handling directly in C
- Powerful relocating macro assembler
- Linker supports linking, relocation, and format generation to produce Flash/PROMable code
- Supports a wide range of industry-standard symbolic formats
- All required ISO/ANSI C libraries are included:
  - character handling, I/O, general utilities
  - string handling, math, and trigonometric
  - low-level routines
- Extended C and EC++ libraries with optimized math and floating-point support
- Single stepping of function calls inside expressions and inside parameter lists
- Complex code and data breakpoints can be set in source code
- Source code can be edited while debugging
- Versatile monitoring of variables, structures, call chain, and locals
- Built-in support for ZiLOG's ZPAKII run control emulator
- Support for hardware debugging via FS2 run control emulator

The IAR Embedded Workbench is a fully Integrated Development Environment for embedded applications. Intelligent tools automate and facilitate development work.

IAR Embedded Workbench™ Professional includes graphical tools for system design, test, and documentation
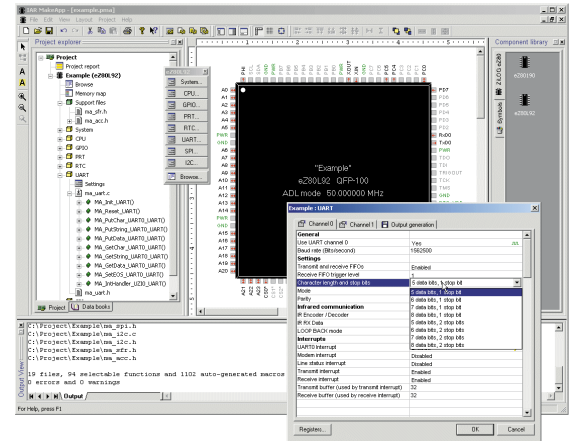
IAR MakeApp™ for ZiLOG's eZ80®

IAR MakeApp™ generates a complete set of customized device driver functions for ZiLOG's eZ80™, including initialization, run-time control, and interrupt handling functions. The drivers are ready to be used with your application software.

Features

• Visual design of device drivers with a simple point-and-click mechanism
• Built-in rule checking detects illegal settings and resource conflicts integration
• Automatic calculation of special function register values
• Optimizing code generator creates efficient and well-tested source code
• Generation of the microprocessor SFR header files and access method macros for the SFR bitfields
• Component browser that visualizes all the details on the chip
• Project report generation with detailed information on chip resources (such as SFRs, pins, and interrupts) as well as the project settings
• Supports the ZiLOG eZ80190 and eZ80L92 devices (support for eZ80F91, eZ80F92, eZ80F93 available Q2 03)
• Supports IAR C/Embedded C++Compiler for eZ80® and ZiLOG's eZ80® C-Compiler

Peripherals

• CPU (Watchdog timer, bus control, RAM control and power control)
• GPIO
• PRT
• RTC
• DMA
• UART
• SPI
• I²C

IAR Embedded Workbench and IAR MakeApp for ZiLOG eZ80® are developed in close cooperation with ZiLOG to ensure that all features in the eZ80® are fully supported.

Device Drivers

• The device drivers are generated in C language
• Optimized driver functions are generated according to project settings. Initialization, run-time control, and interrupt handling functions are generated for each peripheral
• More than 100 device driver functions
• Automatic generation of SFR header files for ZiLOG's eZ80190 and eZ80L92. More than 1,200 macros for eZ80190 and 1,100 macros for eZ80L92 are generated for accessing the SFR bit fields

IAR visualSTATE® for ZiLOG's eZ80®

IAR visualSTATE®, for ZiLOG's eZ80® is a suite of fully integrated tools for the entire development process. It includes a UML-compliant graphical design environment, advanced verification and validation tools, and a very powerful code generator.

When developing with IAR visualSTATE® for ZiLOG's eZ80®, the entire application is based on the design and due to the unique technology it is possible to perform exhaustive testing and generate reliable and production-ready C code – in just a few seconds. The generated code is absolutely consistent with the design; it executes deterministically and can even be more compact than handwritten code.

FL002404-0703