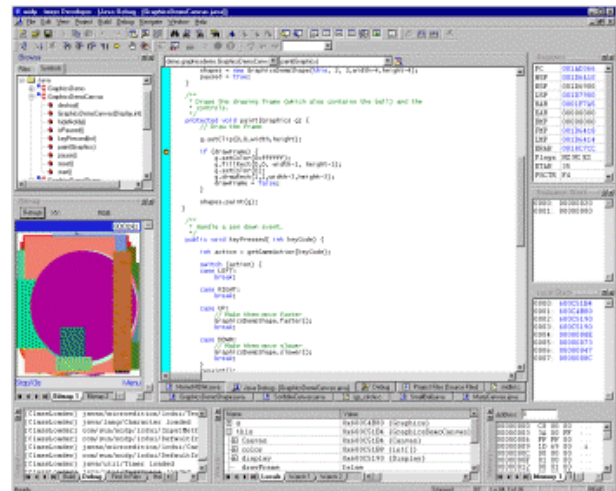


## SNAP-IDE

Integrated Development Environment for the SNAP

- ❑ Developer IDE
- ❑ One year support contract
- ❑ Trace adapter
- ❑ Source code editor
- ❑ On-line manuals and help system
- ❑ C-compiler
- ❑ Assembler
- ❑ Linker
- ❑ Loader and trace/debug tools



### Introduction

**SNAP-IDE** is a visual integrated development environment, optimized for the SNAP. It handles a mix of Java, C, and assembler code. The tools included are source code editor, on-line manuals and help system, C-compiler, assembler, linker, loader and trace/debug tools.

An application project is created, the programs compiled, files are collected and transferred to the target and the target application is run and debugged, using the SNAP-IDE.

As is indicated in the figure, while target debugging is performed, a part of the target memory can be displayed as a bitmap in the SNAP-IDE, e.g. to simulate a customer target LCD, which is perhaps not yet available or connected.

### High and Low level programming

The SNAP-IDE can be used to program applications on all levels. Complex tasks are programmed in Java or C. Time critical parts are programmed in Assembler, i.e. at the machine instruction level.

### Downloading to the Target and Debugging

When an application has been compiled and linked, in the SNAP-IDE, it is downloaded to the target system. The SNAP-IDE is then used to trace or debug the application. Debugging is provided on source code levels, both in C and Java, as well as on machine code level, with all the usual debugging features.

If Java is used, usually Java class files are collected and transferred to a file system in the target. Also data files of different type can be included in the project and transferred.

## Features

Parameter	SNAP-IDE
Java, C source and Assembly-level debugger	<ul style="list-style-type: none"> <li>- <b>Breakpoints, Single-step:</b> Step through the code at Java, C or Assembler level.</li> <li>- <b>Code disassembly:</b> Switch between debugging at Java, C or machine code level.</li> <li>- <b>Watch expressions:</b> Update complex expressions and function calls without stopping program execution.</li> <li>- <b>Java call stack:</b> Displays the top of the Java call stack.</li> <li>- <b>Register window:</b> Control and manipulate registers and flags.</li> <li>- <b>Stack window:</b> Displays the top of the processor stack.</li> <li>- <b>Hex memory dump:</b> Window for display of the content of memory addresses.</li> <li>- <b>STDIO window:</b> Displays Output and input from host PC keyboard.</li> <li>- <b>Bitmap view:</b> Display of the target memory as a bitmap.</li> <li>- <b>Terminal window:</b> Online target debugging through Ethernet or serial line.</li> </ul>
Java VM	<ul style="list-style-type: none"> <li>- 2ME-CLDC, certified by Sun Microsystems</li> <li>- Java.net</li> <li>- Javax.comm.</li> <li>- Subset of com.dalsemi</li> </ul>
C compiler	Ansi C
Moose™ RTOS	Real-Time Operating System that provides threads to be used in C or automatically included as a base for the Java system. Timing is based on the Cjip hardware timeslot, with a frequency of 16kHz. Semaphores are supported, typically used by applications to lock common resources during manipulation.
Board specific drivers with Java and/or C APIs	<ul style="list-style-type: none"> <li>- TCP/IP (HTTP, TFTP, FTP, SMTP, DHCP, UDP, TCP, ICMP)</li> <li>- PPP</li> <li>- RS232</li> <li>- I<sup>2</sup>C</li> <li>- CAN</li> <li>- SPi</li> <li>- 1-Wire</li> <li>- Flash cards</li> <li>- LCD and Touch-panel</li> <li>- Keypad</li> <li>- Digital I/O</li> </ul>
No in-circuit emulator	Advanced simulator that allows application software to be developed and executed without the need of a hardware target.
MIDP	Mobile Information Device Profile (MIDP) is a Sun Microsystems package of standardized Java APIs to create user interface. MIDP provides the core application functionality required by today's standard for mobile information devices (MIDs) such as phones and entry level PDAs.