

AVR[®] ICE 200

User Guide



Table of Contents

Section 1

| | |
|---------------------------------|-----|
| Preface – Read this First | 1-1 |
| 1.1 About this Manual | 1-1 |
| 1.2 Helpful Information | 1-1 |
| 1.3 Tips | 1-1 |
| 1.4 Checklists | 1-1 |
| 1.5 Related Documentation | 1-1 |

Section 2

| | |
|-----------------------------------|-----|
| Introduction | 2-1 |
| 2.1 ICE 200 Features | 2-2 |
| 2.2 ICE 200 Contents | 2-2 |
| 2.3 System Requirements | 2-3 |
| 2.3.1 Hardware Requirements | 2-3 |
| 2.3.2 Software Requirements | 2-3 |
| 2.3.3 Operating Conditions | 2-3 |
| 2.3.4 Host Interface | 2-3 |

Section 3

| | |
|---------------------------|-----|
| General Description | 3-1 |
|---------------------------|-----|

Section 4


| | |
|---|------|
| Using the ICE 200 | 4-1 |
| 4.1 Target Hardware Requirements | 4-1 |
| 4.2 Power and Signal Operating Conditions | 4-1 |
| 4.3 Clock Driver Requirements | 4-2 |
| 4.4 Personality Adapters | 4-3 |
| 4.5 Special Tiny12 Personality Adapter Settings | 4-6 |
| 4.6 Connecting to the Target Application | 4-7 |
| 4.6.1 Checklist | 4-9 |
| 4.7 Configuration | 4-10 |
| 4.8 Quick Start | 4-10 |
| 4.8.1 Checklist | 4-11 |

| | | |
|------------------|--|------|
| 4.9 | Emulator Options Settings | 4-11 |
| 4.9.1 | Device Settings..... | 4-11 |
| 4.9.2 | Clock Selection Settings..... | 4-11 |
| 4.9.3 | Single-step Timers Setting | 4-11 |
| 4.9.4 | EEPROM Restore Setting | 4-12 |
| 4.9.5 | Communication Speed Setting | 4-12 |
| 4.9.6 | Reset Pin Setting (ATtiny12 only)..... | 4-12 |
| <hr/> | | |
| Section 5 | | |
| | Special Considerations | 5-1 |
| 5.1 | External RESET | 5-1 |
| 5.2 | SLEEP Instruction | 5-2 |
| 5.3 | Watchdog Timer (WDT) | 5-2 |
| 5.4 | EEPROM | 5-3 |
| 5.5 | I/O Port Access | 5-3 |
| 5.6 | 16-bit I/O Access (Timer 1 and A/D Converter) | 5-4 |
| 5.7 | UART Data Register | 5-4 |
| <hr/> | | |
| Section 6 | | |
| | Appendix..... | 6-1 |
| 6.1 | Emulating AT90S1200 and ATtiny10/11 | 6-1 |
| 6.1.1 | Using the Include Files | 6-1 |
| 6.1.2 | Using the ATtiny12 Adapter for Emulating the ATtiny10/11 | 6-2 |
| 6.1.3 | Using the AT90S2313 Adapter for Emulating the AT90S1200..... | 6-2 |
| 6.2 | AVR Emulator Chip Errata | 6-2 |
| 6.3 | Troubleshooting | 6-3 |
| 6.3.1 | Feedback and Support | 6-3 |
| 6.4 | Contact Information..... | 6-3 |



Section 1

Preface – Read this First

-
- | | | |
|-------|------------------------------|---|
| 1.1 | About this Manual | <p>This user guide serves as a reference manual for the Atmel AVR[®] ICE 200[™] in-circuit emulator. The AVR ICE 200 User Guide is an easy introduction on how to use the ICE 200, and a detailed reference for advanced users. Throughout the manual, many references to the AVR microcontrollers are made in short form, i.e. AT90S2313 is referred to as S2313 and so on.</p> <p>The user should install the latest version of the AVR Studio available on the Atmel web site.</p> |
| <hr/> | | |
| 1.2 | Helpful Information | <p>This manual contains helpful information to improve the reliability, performance, and longevity of the ICE 200 and the target system.</p> <div style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"><p>NOTICE!</p><p><i>This is a Notice...</i></p></div> <p>Please follow the instructions in a NOTICE carefully.</p> |
| <hr/> | | |
| 1.3 | Tips | <p>Some sections contain useful tips for using the ICE 200. All the tips are emphasized as shown in the example below.</p> <p> Tip! This is a tip!</p> |
| <hr/> | | |
| 1.4 | Checklists | <p>When the detailed descriptions in the <i>Connecting to the Target Application</i> and in the <i>Configuration</i> sections have been used and you are beginning to feel comfortable with the use of the ICE 200, you can use the checklists at the end of these sections for fast setup of a new project. The checklists are of great help for getting the debugging system online without problems. However, novice users should also check that the operating conditions of the target system are compliant to the requirements of ICE 200. This is described in the <i>Using the ICE 200</i> section.</p> |
| <hr/> | | |
| 1.5 | Related Documentation | <p>The Atmel CD-ROM contains various documentation relating to the use of AVR microcontrollers and of the debugging tools including AVR Studio User Guide, AVR Assembler User Guide and complete microcontroller datasheets.</p> |



Section 2

Introduction

The ICE 200 in-circuit emulator provides an easy way of debugging embedded systems that utilizes the Atmel AVR microcontroller. It emulates 11 different devices of the AVR and the Tiny AVR families.

The philosophy of the ICE 200 is to provide an easy-to-use debugging platform, with a minimum of differences between the emulator and the actual processor it is emulating. The AVR emulator chip used by the ICE 200, is produced in the same process technology as the microcontroller it is emulating. This provides identical electrical characteristics. On-board debugging resources ensure non-intrusive software emulation. The ICE 200 hardware also includes an automatic configuration system that makes the process of connecting the target to the emulator an easy task.

Figure 2-1. The ICE 200 Components



When used with the AVR Studio debugging environment, the ICE 200 gives the user full run time control, unlimited number of breakpoints, symbolic debugging and full memory and register visibility. Multiple ICE 200 emulators can be used by AVR Studio at the same time, only limited by the number of serial ports available, giving a high degree of flexibility.

-
- 2.1 ICE 200 Features**
- Devices Supported
ATtiny12, AT90S2313, AT90S2333/4433, AT90S4414/8515, AT90S4434/8535, ATtiny10/11 (using ATtiny12 adapter), AT90S1200 (using AT90S2313 adapter)
 - Supports 8 MHz (+4.0V to +6.0V) AVR Emulator Chip (varies between devices being emulated)
 - Wide Target Voltage Range (+2.7V to +5.5V)
 - Emulator Chip Provides Excellent AC Characteristics
 - Non-intrusive
 - Target Voltage Sensing Ensures Secure Operation
 - Personality Adapter for Each of the Supported Processors
 - 32-bits Cycle Counter
 - I/O Continues to Operate in Halt State After a Break or Breakpoint
 - Single-stepping or Continuous Timer Operation while Single-stepping Code. Utilizes the AVR Studio Debugging Environment that adds: Full Run Time Control: run, break, trace into, step over, step out, run-to-cursor, reset, autostep and multistep
 - Unlimited Number of Breakpoints
 - Symbolic Debugging Support
 - Full Visibility of and Access to register File, SP, PC and Memories
 - Access to all I/O Registers – See Section 5: Special Considerations
 - Auto Log Points – Non-real Time Logging/Watches

-
- 2.2 ICE 200 Contents**
- The ICE 200 contains the following items:
- ICE 200 Main Board, pod and two Flexible Printed Circuit Cables
 - Personality Adapters for:
 - ATtiny12 (8-pin DIP)
 - AT90S2313 (20-pin DIP)
 - AT90S2333/4433 (28-pin DIP)
 - AT90S4414/8515 (40-pin DIP)
 - AT90S4434/8535 (40-pin DIP)
 - 9-pin RS232C Cable
 - Atmel CD ROM containing:
 - AVR data books
 - Application notes
 - AVR Studio
 - AVR Assembler
 - ICE 200 User Guide (this document)
 - Power Cable
 - Diagnostic Adapter for Test Purposes

2.3 System Requirements

- 2.3.1 Hardware Requirements**
- Pentium-class personal computer with the following specifications is recommended:
- 16M Byte RAM, or more
 - 3M Byte of free hard disk space
 - CD-ROM or Internet access (for software and data books)
 - VGA monitor, or better
 - 16650 Compatible Serial Port (COM port)

- 2.3.2 Software Requirements**
- The following operating systems are currently supported by AVR Studio:
- AVR Studio v2.00 or later installed. See the Atmel web site (www.atmel.com) for latest version.
 - Microsoft Windows NT 3.51
 - Microsoft Windows NT 4.0
 - Microsoft Windows 95
 - Microsoft Windows 98

Note: AVR Studio will be updated to execute new versions of these operating systems. See AVR Studio User Guide for latest information.

- 2.3.3 Operating Conditions**
- Operation Temperature: 0°C - 70°C
 - Operating Humidity: 10 - 90% RH (non-condensing)
 - Supply Voltage: +9.0V to +12.0V DC or 9.0V AC
 - Supply Current: 400 mA

NOTICE!

Violating the recommended operating conditions for the ICE 200 might cause incorrect operation and damage the emulator.

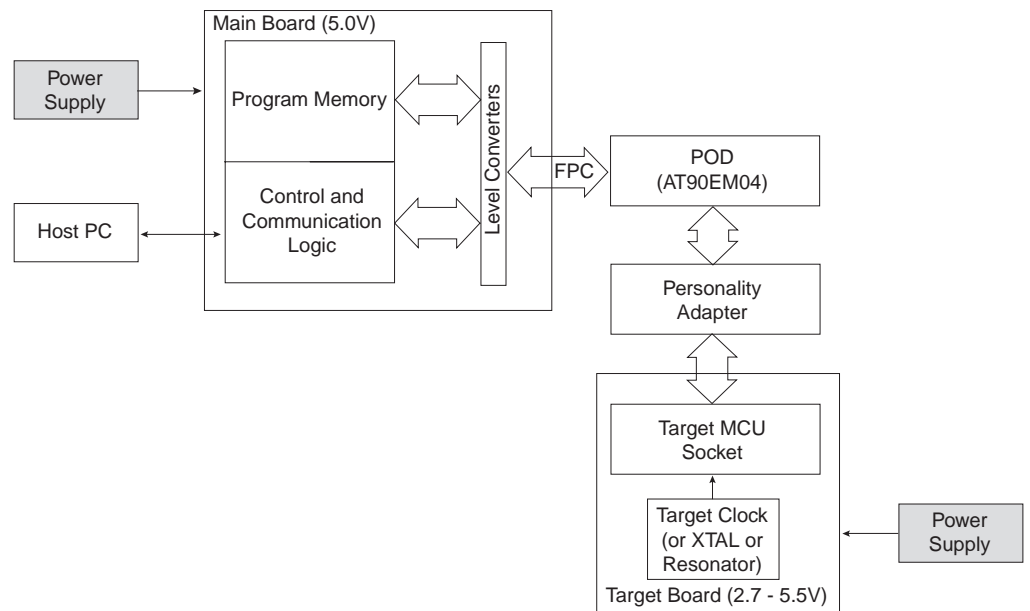
- 2.3.4 Host Interface**
- RS-232C @ 19200 bps, 1 start-, 8 data- and 1 stop-bit, no parity. 9-pin female connector.

Section 3

General Description

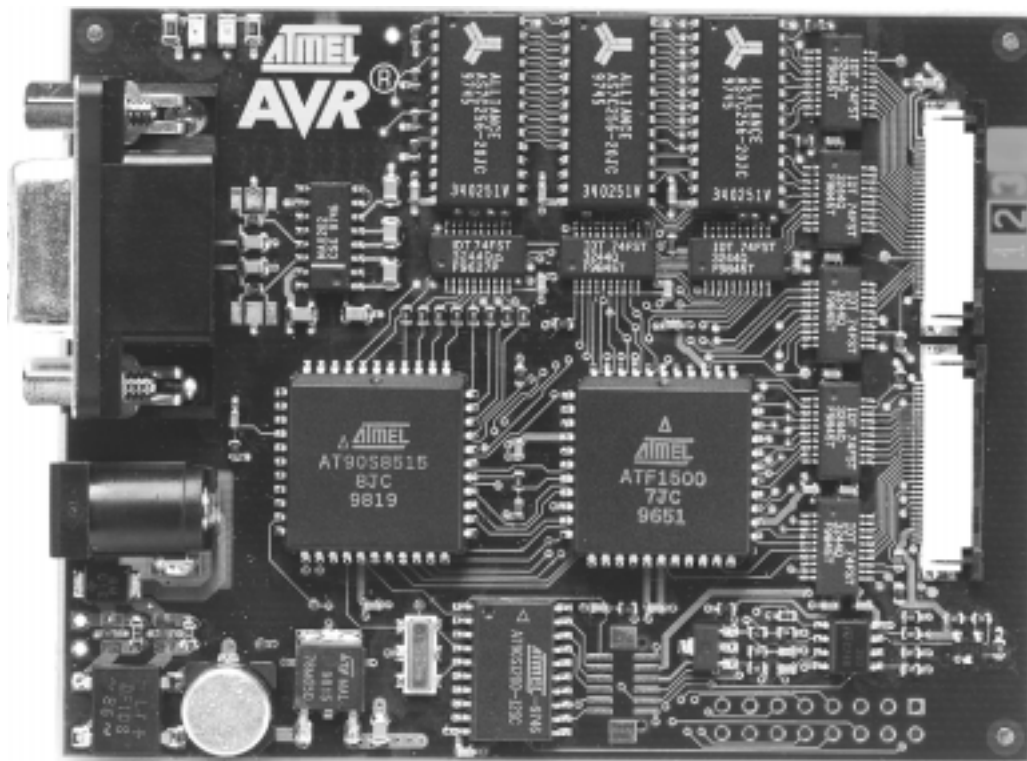
Figure 3-1 shows a simplified block diagram of the ICE 200 connected to a target board (the application). Power supplies and a host PC are also shown.

Figure 3-1. ICE 200 – Simplified Block Diagram



The main board (Figure 3-2) contains the program memory (overlay memory) which holds the application code that is being emulated. The main board also contains logic for communicating with the host PC, and the breakpoint logic. The level converters allow the target to operate at a different supply voltage from that of the emulator. The level converters also protect the emulator and the target from being damaged if only one of them is powered. Due to this feature, a strict power-up sequence is not required.

Figure 3-2. ICE 200 – Main Board



The FPC or Flexible Printed Cable (Figure 3-3) connects the main board to the ICE 200 Pod. The actual appearance of the FPC may differ from the figure.

Figure 3-3. ICE 200 – FPC



NOTICE!

The Flexible Printed Cable must not be folded.

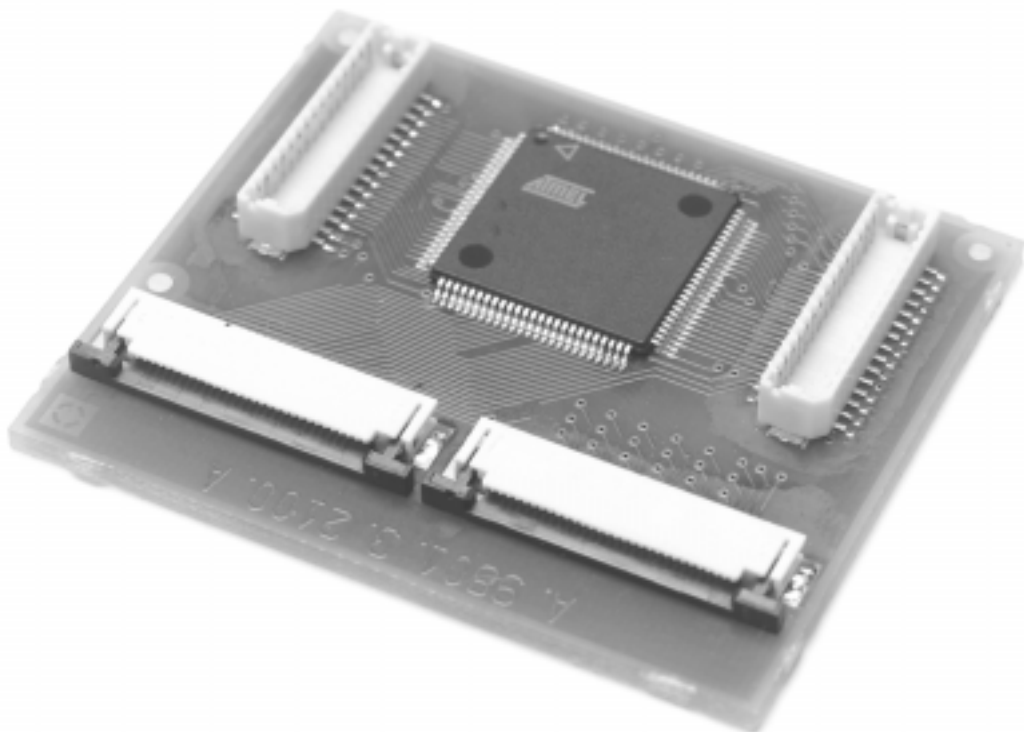
NOTICE!

Do not disassemble the Flexible Printed Cable from the pod or ICE 200 main board.

General Description

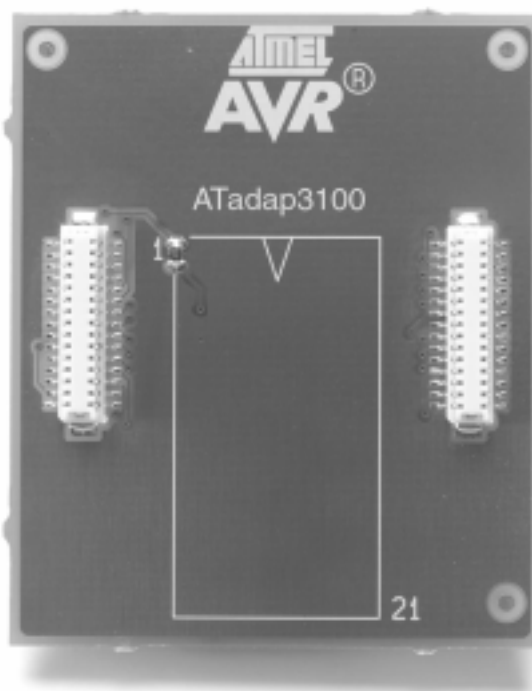
The pod (Figure 3-4) contains the AVR emulator chip. Note that the AVR emulator chip must be supplied with power and a clock source, i.e. a crystal, resonator, oscillator or any other clock generator, from the target.

Figure 3-4. The ICE 200 Pod



The personality adapters (Figure 3-5) map the pinout from ICE 200 Pod to each microcontroller it supports. The adapters include an identification code that AVR Studio uses for automatic device type detection. The ICE 200 kit contains five different personality adapters for dual-in-line package devices.

Figure 3-5. The ICE 200 – Personality Adapter for the AT90S8535 – ATadap3100





Section 4

Using the ICE 200

Before opening the ESD protection bag, take precaution to eliminate electrostatic discharge. Always use ESD protected tools and clothing when using the ICE 200. Grounded wrist-band and static-dissipative work surface provides the most efficient ESD protection. The ICE 200 should be handled with the same care as any CMOS component.

NOTICE!

ESD (Electrostatic Discharge) SENSITIVE DEVICE. Do not use the ICE 200 outside an ESD protected environment.

A discharge may result in permanent damage or performance degradation.

4.1 Target Hardware Requirements

The target application hardware must include both power supply and a clock source. The ICE 200 can not function unless these conditions are met. Note that the emulator also supports the internal RC-oscillator option to the Tiny12 device.

Tip!

You can use an AVR development board (ATMCU00100 or ATSTK200) for using the ICE 200 as a standalone emulator platform.

Please follow the recommended operating conditions listed in the next two sections. These conditions also apply for the standard AVR microcontrollers.

4.2 Power and Signal Operating Conditions

Table 4-1. Recommended Operating Conditions, Power and Signals ($T_A = 0^\circ\text{C}$ to 70°C , $V_{CC} = +2.7\text{V}$ to $+5.5\text{V}$, $\text{GND} = 0\text{V}$)

| Symbol | Min | Max |
|---------------------------------------|----------|--------------------------|
| V_{CC} | 2.7V | 5.5V |
| AV_{CC} | V_{CC} | $V_{CC} \pm 0.3\text{V}$ |
| AGND | GND | GND |
| AREF | AGND | AV_{CC} |
| V_{SI} (Signal Input Voltage) | -0.5V | $V_{CC} + 0.5\text{V}$ |
| V_{RESET} (RESET pin input Voltage) | GND | V_{CC} |

Note: When $V_{CC} < 2.4\text{V}$, the AVR emulator chip is reset and the program memory disconnected.
The ICE 200 does not support +12V RESET pin voltage that is used for parallel programming.

4.3 Clock Driver Requirements

AVR microcontrollers are fully static designs. The processor clock can be stopped externally. The AVR emulator chip needs a clock to communicate with the main board. Without a clock source, the host PC gets a serial communication time-out when reading status or variables from the emulator. Please refer to the datasheet for information about clock oscillator options.

Table 4-2. Recommended Operating Conditions, Clock Drive (+4.0V to +5.5V) ($T_A = 0^\circ\text{C}$ to 70°C , $V_{CC} = +4.0\text{V}$ to $+5.5\text{V}$, GND = 0V)

| Symbol | Min | Max |
|-----------------------------------|------------|--------|
| f_{OSC} (ATtiny12) | 32.768 kHz | 8 MHz |
| f_{OSC} (AT90S2313) | 32.768 kHz | 10 MHz |
| f_{OSC} (AT90S4433/2333) | 32.768 kHz | 10 MHz |
| f_{OSC} (AT90S8515/4414) | 32.768 kHz | 8 MHz |
| f_{OSC} (AT90S8535/4434) | 32.768 kHz | 8 MHz |

Table 4-3. Recommended Operating Conditions, Clock Drive (Low-voltage +2.7V to +4.0V) ($T_A = 0^\circ\text{C}$ to 70°C , $V_{CC} = +2.7\text{V}$ to $+4.0\text{V}$, GND = 0V)

| Symbol | Min | Max |
|-----------------------------------|------------|-------|
| f_{OSC} (ATtiny12/) | 32.768 kHz | 4 MHz |
| f_{OSC} (AT90S2313) | 32.768 kHz | 4 MHz |
| f_{OSC} (AT90S4433/2333) | 32.768 kHz | 4 MHz |
| f_{OSC} (AT90S8515/4414) | 32.768 kHz | 4 MHz |
| f_{OSC} (AT90S8535/4434) | 32.768 kHz | 4 MHz |

NOTICE!

Using the ICE 200 outside the recommended operating conditions will cause incorrect operation and can damage the emulator.

4.4 Personality Adapters

The ICE 200 is supplied with five different personality adapters. Each adapter makes the pinout mapping for one or more AVR microcontrollers.

Tip!

Mounting a DIP socket on the personality adapter reduces the risk of breaking pins on the adapter, thus extending adapter lifetime.

Tip!

You can mount additional DIP sockets to the adapter to increase the space between the adapter and the target. However, the number of extra sockets should be kept at a minimum.

If you are utilizing surface mount device (SMD) versions of the supported AVR microcontrollers, you need to obtain an SMD adapter that converts from DIP to the appropriate socket.

Figure 4-1. Personality Adapter for ATtiny12 – ATadap3400

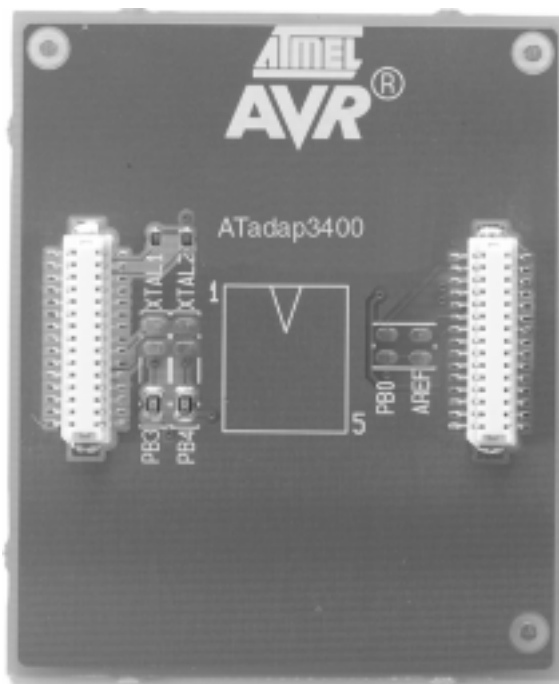


Figure 4-2. Personality Adapter for AT90S2313 – ATadap3300

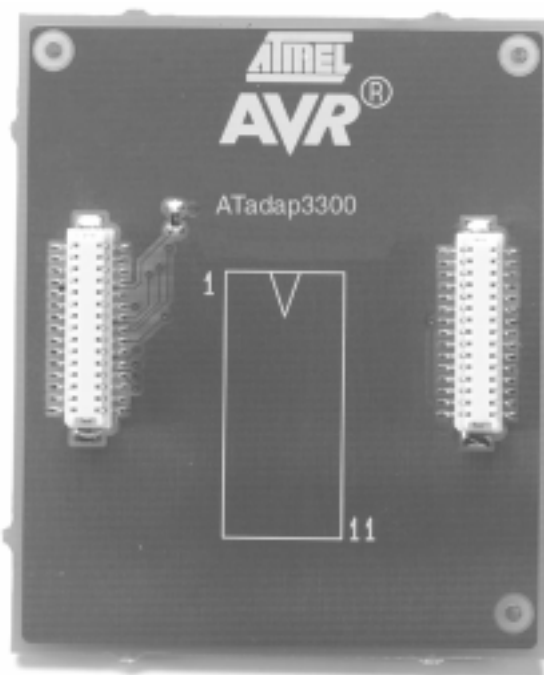


Figure 4-3. Personality Adapter for AT90S4433/2333 – ATadap3200

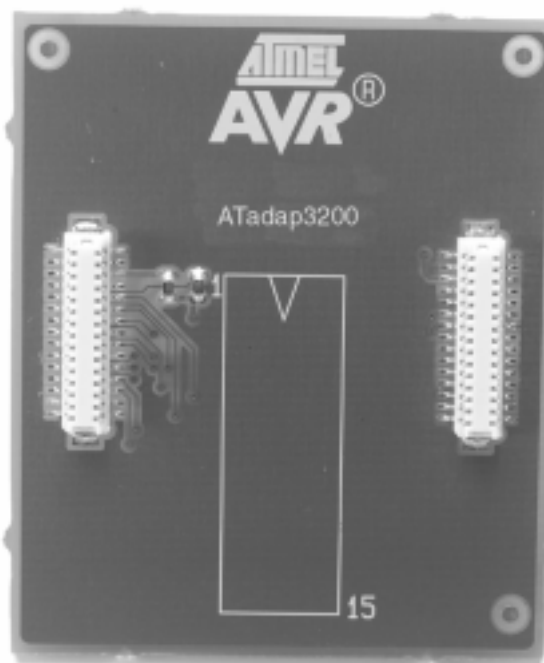
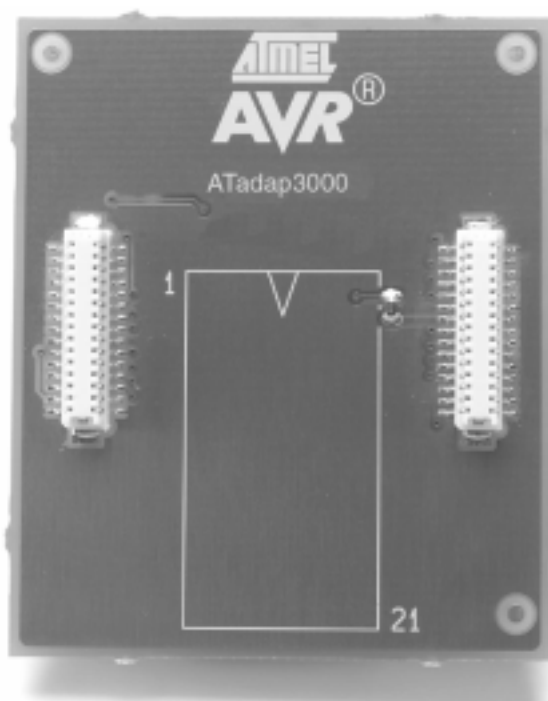


Figure 4-4. Personality Adapter for AT90S8515/4414 – ATadap3000**Figure 4-5.** Personality Adapter for AT90S8535/4434 – ATadap3100**NOTICE!**

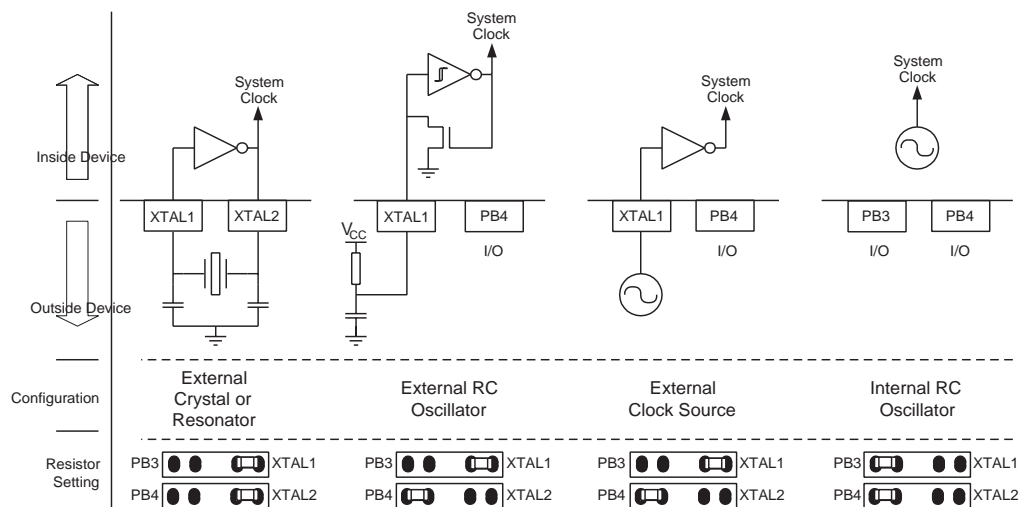
Do not change personality adapter without turning power-off on both the emulator and the target.

4.5 Special Tiny12 Personality Adapter Settings

The AVR ATtiny12 microcontroller includes some special oscillator pin features that could not be implemented in the AVR emulator chip due to its multiple device support. However, the options are supported on the personality adapter by changing the position of two zero ohm resistors that configure the port and the oscillator pin mapping. The setting of the resistors is shown in Figure 4-6.

The resistor setting depends on the desired clock configuration. Select the resistor setting based on the table below. Default factory resistor setting is internal RC oscillator.

Figure 4-6. Settings of Resistors on ATadap3400



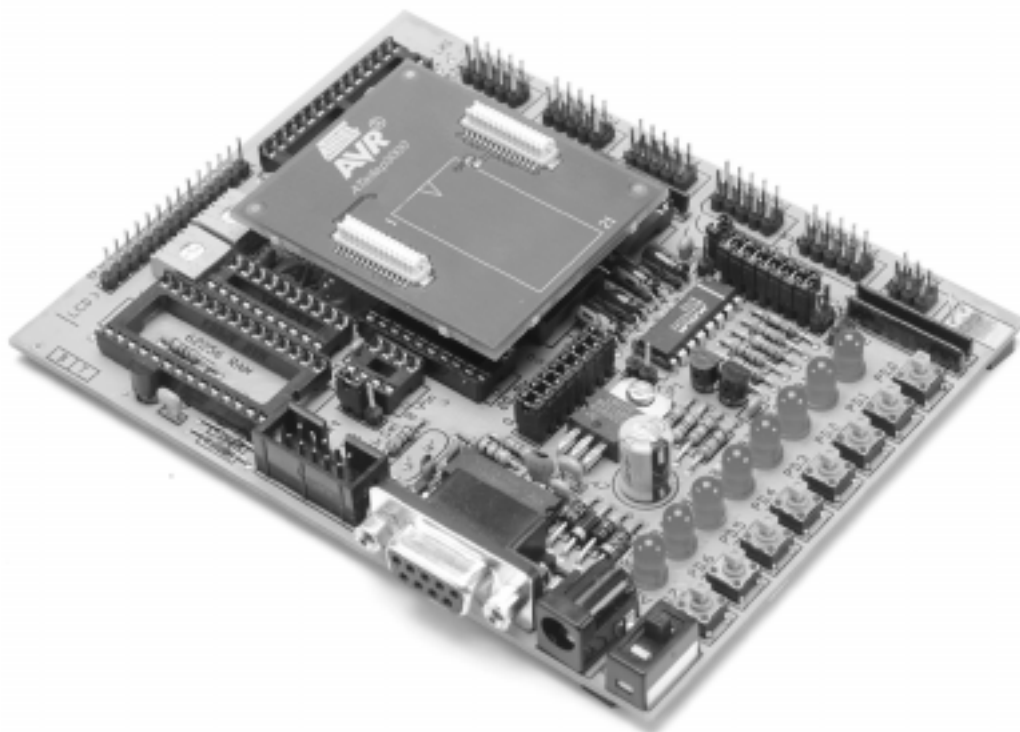
The personality adapter board has four additional resistors. These are used as identification codes for the automatic configuration and for production test purposes. Do not remove these resistors!

4.6 Connecting to the Target Application

The procedure in this section must be followed when setting-up the emulator. The end of this section contains a useful checklist for ensuring fast and safe installation of the system.

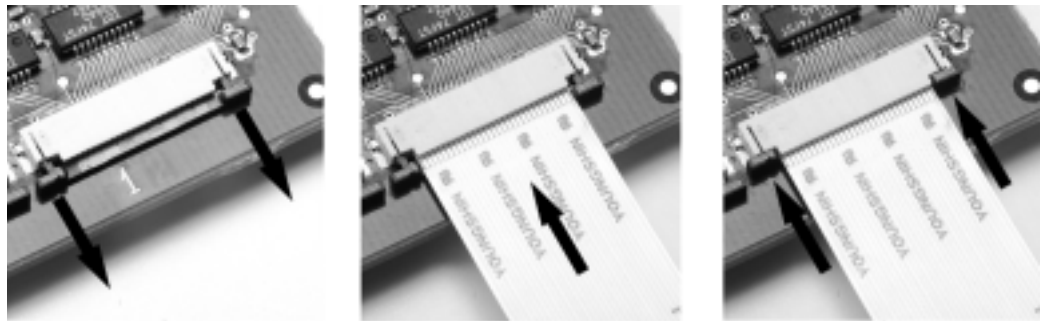
1. Before connecting the ICE 200 to the target application, make sure that the ICE 200 and the target application are not powered. This also applies when the ICE 200 is removed from the target. When connecting or disconnecting the ICE 200 from the host PC, make sure that neither the ICE 200 nor the target application is powered.
2. Start inserting a personality adapter (see Figure 4-7). Make sure that pin 1 on the personality adapter corresponds with pin 1 on the target socket.

Figure 4-7. Inserting a Personality Adapter into the Target Hardware



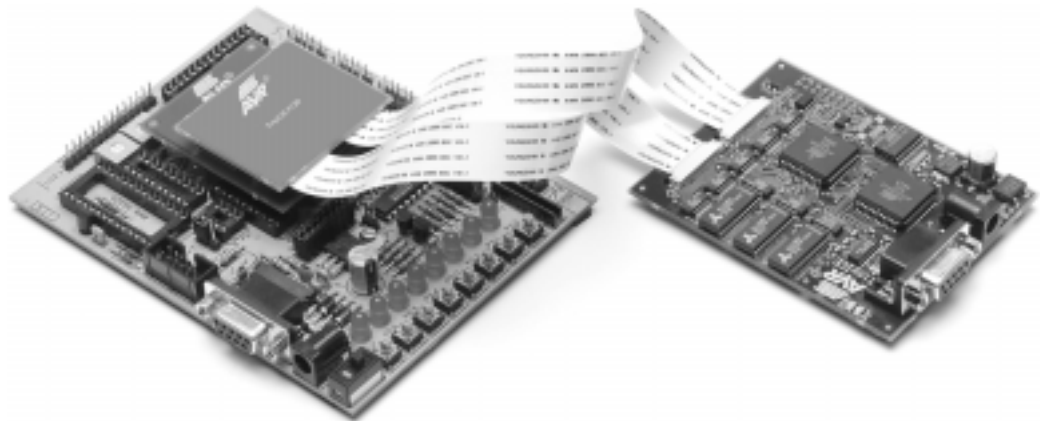
3. If the flexible printed circuit cable (FPC) is not already connected to the pod and the main board, connect it as shown in Figure 4-8. Note that the FPC lead prints must face up.

Figure 4-8. Mounting the FPC to the pod and the Main Board



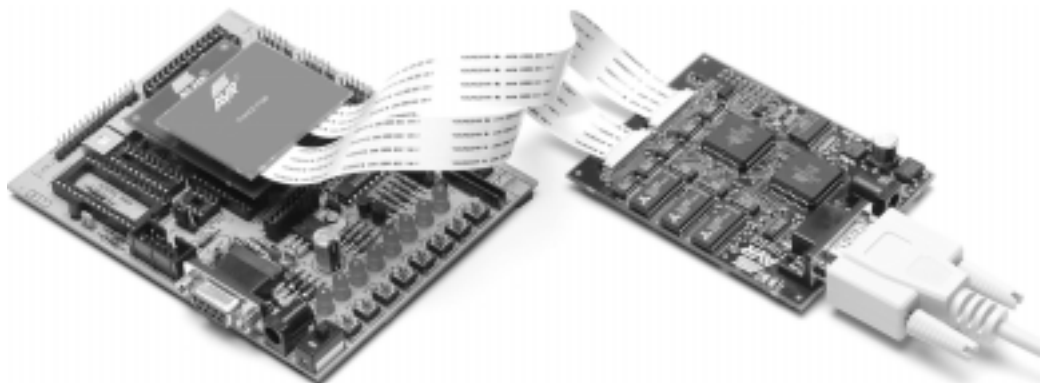
4. Mount the pod onto the personality adapter as shown in Figure 4-9. Do not use force since the pod only fits one way into the personality adapter.

Figure 4-9. Mounting the pod onto the Personality Adapter



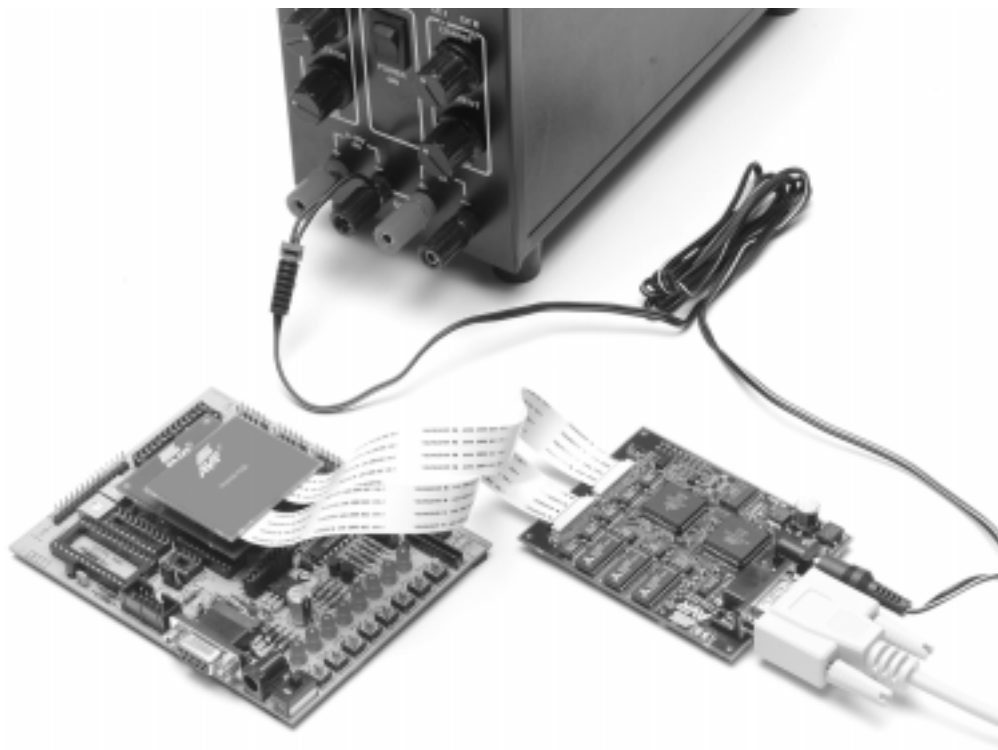
5. Connect the ICE 200 to the host PC. Use the 9-pin RS232C cable that is shipped with the ICE 200. Connect the male cable connector to the ICE 200 and the female cable connector to the host.

Figure 4-10. Connecting the ICE 200 to a Host PC



6. AVR Studio can now be started. However, do not open any files before connecting power supply to the ICE 200. If the ICE 200 is not powered then AVR Studio cannot detect the emulator and therefore enters simulation mode.
7. The ICE 200 has no power switch. Just connect the power supply cable shipped with the ICE 200 to a +9V to +12V DC or a 9V AC power supply, and then connect the power cable to the ICE 200 (see Figure 4-11). A battery eliminator is a good alternative to the laboratory power supply shown on the figure.

Figure 4-11. Connecting Power Supply to ICE 200



8. Enable the target power supply. The red LED will now be lit, telling that power is present, but no connection to the host PC has been established.
- The hardware is now ready for use. Use the checklist below to ensure that the setup was done correctly, and then proceed to the next section: Configuration.

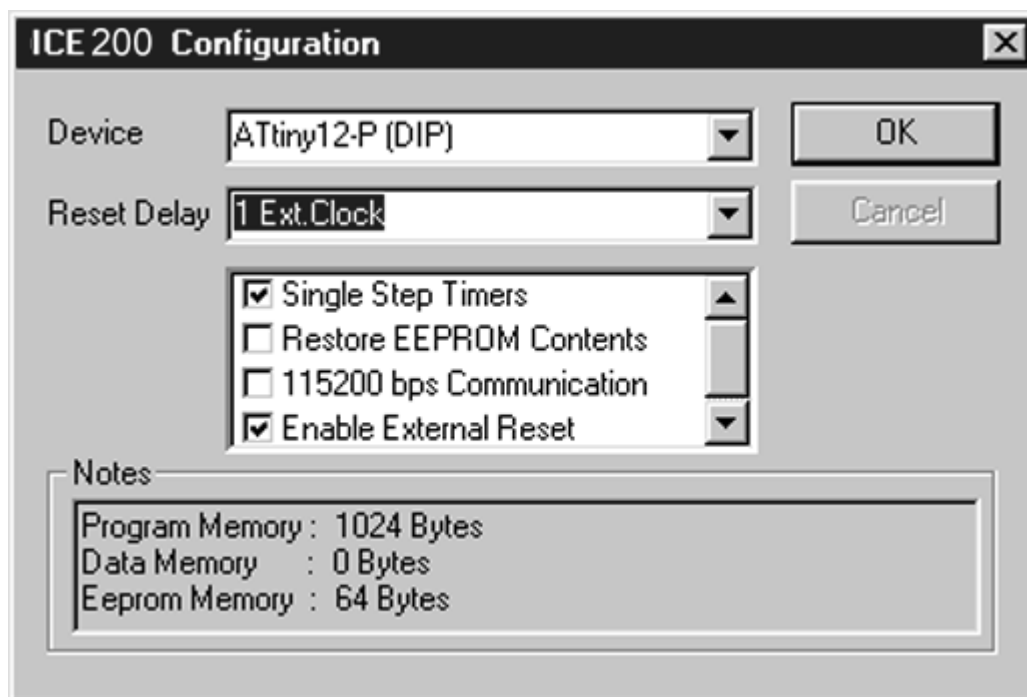
4.6.1 Checklist

1. ☐ Turn-off power on all units.
2. ☐ Insert personality adapter.
3. ☐ Pins on adapter agree with pins on target.
4. ☐ Mount pod.
5. ☐ Mount FPC (first time only, do not disassemble).
6. ☐ Connect RS232C cable.
7. ☐ Turn-on power on all units.

- 4.7 Configuration** When the ICE 200 is connected to the target application, the next step is to set the device configuration of the part you are using. This is required when an application code project is opened for the first time, but can later be changed in the emulator options menu. The configuration is stored in a separate file, [project name].avd, in the same directory as the application code.

- 4.8 Quick Start** Follow the procedure described below to configure the ICE 200.
1. Connect the ICE 200 and start AVR Studio as described in previous section.
 2. Open the object file. If the file is opened in AVR Studio for the first time, the Emulator Options Window automatically appears, see Figure 4-12. The red LED will now be turned off and the green will be turned on (if it is not already on), indicating that the connection between the ICE 200 and the host PC is established.

Figure 4-12. ICE 200 Configuration Dialog



3. Set up the desired configuration. A detailed description is found in the following sections, however, the default settings are sufficient in most cases due to the automatic personality adapter detection.
4. Press the OK button.
5. The ICE 200 is now ready for use!

Tip!

To change configuration for the current project, select the Option – Emulator Options menu.

For advanced users, Section 4.9 describes in more detail the possible settings in the Options menu.

The software is now ready for use. Use the checklist below to ensure that the setup was done correctly. The ICE 200 is now ready for debugging.

🔗 **Tip!**

If AVR Studio cannot connect to the emulator, make sure no other programs are using the serial port.

4.8.1 Checklist

1. ☐ The personality adapter selected corresponds to the device that is to be used.
2. ☐ Target application is correctly connected to the emulator (as described in previous section).
3. ☐ AVR Studio detects that the emulator is present when opening the application code.
4. ☐ The Options menu for ICE 200 was shown when opening the source file (only first time).
5. ☐ The device in the Options menu correspond to the chip configuration you want to use.
6. ☐ The text "AVR Emulator" appears in the status bar of AVR Studio found in the lower right corner of the window.

4.9 Emulator Options Settings

4.9.1 Device Settings

Some devices have identical pinout and functionality. An example is the AT90S8515 and the AT90S4414 devices, there is no need for a personality adapter for each of them. The automatic personality adapter detection ensures correct pinout configuration of the AVR emulator chip. When selecting a specific device enter, the emulator options menu and select the device from the device list.

4.9.2 Clock Selection Settings

Many AVR microcontrollers have fuse bits for selecting the reset delay time. The reset delay is necessary for the clock oscillator to stabilize. The time it takes for the oscillator to stabilize depends on the crystal or the resonator. If an external clock source is used then the reset delay can be set to only a few clocks. The reset delay fuse bits (CKSEL/FSTRT depending on device) can be set or cleared by a parallel or serial programmer in an actual device. In the emulator, they are set in the options menu.

For more information about the reset delay fuses, please refer to the datasheets.

4.9.3 Single-step Timers Setting

This setting allows single-stepping of the timers if checked. If cleared the timers continue to count (if enabled) even after the program execution is stopped by a user break or breakpoint. All other peripherals (SPI/UART/EEPROM/PORTs) continue to operate when the program execution is stopped.

This feature allows cycle-by-cycle debugging of the counter value, which is useful for event timing. However, in many cases, stopping the counter operation while debugging might not be desired. One example is when the timer is used in PWM mode. Stopping the timer in this case might damage the equipment that is being controlled by the PWM output.

Note that eventual timer interrupts will not be handled before execution is resumed.



4.9.4 EEPROM Restore Setting

Some AVR devices have on-chip EEPROM data memory. The ICE 200 emulates the EEPROM by using an SRAM replacement inside the AVR emulator chip. This is done to eliminate problems with EEPROM write endurance. However, by doing so, a new problem is introduced since a power loss on the target will result in loss of the data stored in the SRAM that emulates the EEPROM.

A split solution handles the power loss situation. First select the EEPROM restore option. Before removing the power, take a snapshot of the EEPROM contents by pressing the EEPROM snapshot button (Figure 4-13). This will tell the ICE 200 main board to read all EEPROM data into a buffer. When power is switched off and then on again, the ICE 200 will restore the contents of the buffer to the SRAM before starting code execution.

Figure 4-13. EEPROM Snapshot Button



4.9.5 Communication Speed Setting

The default communication speed between AVR Studio and the ICE 200 is 19200 bps. If the option “115200 bps Communication” is selected, the communication speed is changed to 115200 bps. Some systems cannot handle this speed, so if AVR Studio loses contact with the emulator after enabling this option, it should be disabled.

4.9.6 Reset Pin Setting (ATtiny12 only)

When selecting ATtiny12 an additional option selection appears in the emulator options dialog box. The ATtiny12 device has a programmable fuse that lets the RESET pin function as an input pin (PB5). When this option is selected, PB5 functions as a normal reset pin. When this option is not selected, PB5 functions as an input pin. The device is then only reset at power-on or when giving a reset command from AVR Studio. Refer to the ATtiny12 datasheet for more information.

Section 5

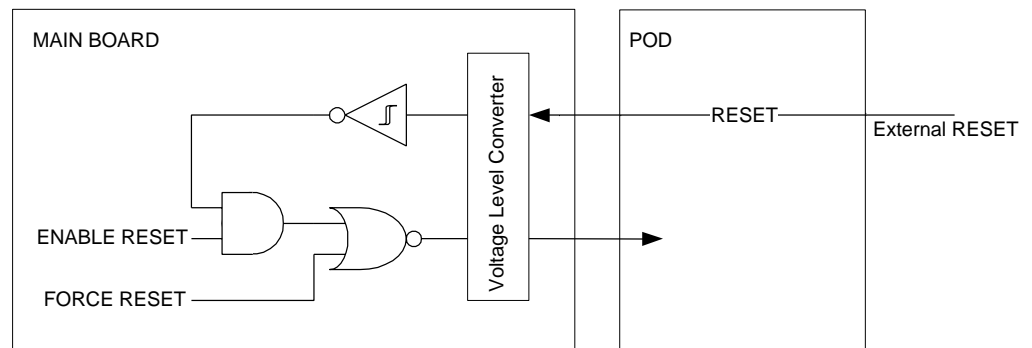
Special Considerations

The ICE 200 accurately emulates most AVR features. However, there are some differences worth noting. Most of the exceptions apply to controlling the program flow, i.e. single-stepping and so on. Program flow control is an extension to the normal functions of the microcontroller that allows the user to do the debugging. This extension must not interfere with the normal program execution (run mode). If the program execution is stopped (stopped mode) and then restarted, or the program is executed line-by-line (single-stepping), the program functionality can, in some cases, be affected.

5.1 External RESET

The ICE 200 main board has to be able to control the reset pin on the AVR emulator chip. An external reset source must therefore go via the control logic as shown on Figure 5-1. This is handled automatically by the pod and main board

Figure 5-1. ICE 200, External RESET Circuit



The main board is working with 5V supply and the pod uses the target voltage. Therefore, a level converter is inserted between the two systems. The extra logic and the level converters introduce a small, and for most of the time, negligible delay. Note that the voltage converters do not handle a +12V input voltage on the RESET pin which is used for enabling the parallel programming on standard parts.

After a power-up, the reset is forced active while configuring the AVR emulator chip, introducing a 1 - 10 ms delay.

5.2 SLEEP Instruction

If a SLEEP instruction is executed and sleep is enabled the AVR emulator chip will enter one of the AVR low-power modes (power-down, power-save or idle). The power consumption will be slightly higher compared to the real chip due to the higher complexity of the emulator chip. In room temperature (25°C) this is not significant.

Tip!

When debugging your application, you can replace the SLEEP instruction with a NOP by using macros.

IAR C example:

```
#ifdef SLEEPEMU
#define SLEEP() _SLEEP()
#else
#define SLEEP() _NOP()
#endif
```

5.3 Watchdog Timer (WDT)

The watchdog timer operates asynchronously (it has its own clock) to the rest of the system. The watchdog timer reset instruction is therefore repeatedly issued in stopped mode to avoid false resets if the watchdog timer is enabled. Debugging the exact timing of the WDT behavior while doing single-stepping or when stopping the program execution, is therefore not supported by the ICE 200.

Disabling WDT is secured by the WDTTOE (WDT turn-off enable) bit. Following is an assembly program example that shows a WDT disable sequence:

disableWDT:

```
( cli ) ; (only needed if any interrupts are in use)

ldi r16, (1<<WDTTOE)+(1<<WDE) ; Set the WDT turn-off enable bit and
                                ; the WDE bit

out WDTCR, r16

ldi r16, (0<<WDE) ; Within 4 cycles, clear the WDEbit
out WDTCR, r16

done:

( sei ) ; (only needed if any interrupts are in use)
```

The four cycle time-out for the WDT disabling is not supported by the AVR emulator chip when in stopped mode. This means that single-stepping this sequence will not disable the WDT.

Tip!

You can use the run to cursor to the instruction after the WDT enable bit is cleared (labelled done in the example above), instead of doing single-stepping. Or use the following macro (for AVR assembler only).

MACRO:

```
.macro disableWDT
( cli )

ldi r16, (1<<WDTTOE)+(1<<WDE)
out WDTCR, r16

ldi r16, (0<<WDE)
out WDTCR, r16

( sei )
.endmacro
```

USAGE:

```
disableWDT
```



5.4 EEPROM

When writing to the on-board EEPROM on an AVR device, a special sequence must be used to ensure the integrity of the EEPROM data. Following is an assembly program example that shows an EEPROM write sequence:

```
writeEEPROM:
    ( cli )                      ; (only needed if any interrupts are in use)
    sbi    EECR, EEMWE
    sbi    EECR, EWE
done:
    ( sei )                      ; (only needed if any interrupts are in use)
```

The four cycle time-out for the EEPROM write is not supported by the AVR emulator chip when in stopped mode. This means that single-stepping this sequence will not strobe an EEPROM write.

Tip!

You can use the run to cursor to the instruction after the EEPROM write enable bit is set (labeled done in the example above), instead of doing single-stepping. Or use the macro defined below.

MACRO:

```
.macro writeEEPROM
    ( cli )
    sbi    EECR, EEMWE
    sbi    EECR, EWE
    ( sei )
.endmacro
```

USAGE:

```
writeEEPROM
```

5.5 I/O Port Access

A special situation occurs when single-stepping a change of the PORT value as shown in the following example:

```
write_with_readback:
    ldi    r16, 0xFF              ; Set all pins as output
    out    DDRx, r16              ; --
    out    PORTx, r16             ; Set the PORTx values
    in     r16, PINx              ; Read the PINx values
    in     r17, PINx              ; Read the PINx values
```

When running this example program at full-speed in the ICE 200 or in a real chip, the value read back in r16 will not end up being the value written at the first line, but will contain the value the port pins had the cycle before the port was written. This is the correct behavior. The PINx value must be synchronized, and therefore it is delayed one cycle to avoid erratic port behavior caused by metastability. r17 will therefore contain the value written to the port. However, when the program is single-stepped, the value of the PINx will change immediately after the single-step and the value of r16 will contain the same value as before.

Changing pin values on an I/O port from AVR Studio when the ICE 200 emulator is stopped does not represent any problem. However, note that, as for the single-stepping case, the pin values are changed immediately.

Clearly this is not a real problem, but it is important to be aware of the effects of the two cases described above. If not, an incorrect program might seem to work in the emulator, but will not work in the real chip.



5.6 16-bit I/O Access (Timer 1 and A/D Converter)

Reading or writing 16-bit values directly from AVR Studio can cause some problems. To read, for example the counter value from timer 1, a 16-bit value, one of the bytes must be stored in a temporary register. This temporary register will be corrupted if the 16-bit value is read when the program execution is stopped.

Tip!

Using the following macros (for AVR assembler only) will solve the 16-bit access problem when using symbolic debugging.

MACROS:

```
.macro outw
    (cli)
    out    @2, @0
    out    @2-1, @1
    (sei)
.endmacro
```

```
.macro inw
    (cli)
    in     @2, @0-1
    in     @1, @0
    (sei)
.endmacro
```

USAGE:

```
inw      r17, r16, TCNT1H           ; Reads the counter value
outw      TCNT1H, r17, r16          ; Writes the counter value
```

When using symbolic debugging in C, the entire C line is executed for each set. Therefore the 16-bit read or write problem will not occur in this situation.

5.7 UART Data Register

Reading the UART Data Register cleans the RXC bit in the UART Control Register. Hence, the monitor program does not attempt to read the UART Data Register. Therefore, the value displayed by AVR Studio for this register does not reflect the real value of this register.



Section 6

Appendix

6.1 Emulating AT90S1200 and ATtiny10/11

6.1.1 Using the Include Files

Always use include files for the I/O registers addresses and for bit definitions in your source code files. This will ease the process of porting code from one microcontroller to another. The files can be found on the CD-ROM which is included in the ICE 200 kit. Copy the include file to your project directory, and include it in the top of the program code as shown below:

(AVR Assembler example)

```
.include "l200def.inc"
```

Then, when writing a value to a I/O register, use the following notation:

(AVR Assembler example)

```
ldi    r16, (1<<DDD1) + (1<<DDD4)    ; Set port D pin 1 and 4 as output and
                                         ; the rest as input.

out     DDRD, r16
```

Note the use of the bit definitions. DDD1 and DDD4 are pin definitions in form of bit position, and therefore they must be shifted this number of bits to the left to make a correct mask.

Interrupt vector locations might differ from part-to-part. This is easily handled by using the vector definitions found in the include files.

(AVR Assembler example)

```
.include "l200def.inc"

.org    0
rjmp    RESET_Handler

.org    INT0addr
rjmp    INT0_Handler

.org    OVF0addr
rjmp    OVF0_Handler

.org    ACIaddr
rjmp    ACI_Handler

... ( program code starts here )
```

6.1.2 Using the ATtiny12 Adapter for Emulating the ATtiny10/11

The ATtiny10 and 11 are both subsets of ATtiny12. Therefore, it is possible to select the ATtiny12 device when configuring the ICE 200 to support either ATtiny10 or ATtiny11. These devices all have the same pinout, but ATtiny10/11 does not have the following features:

- **Brown Out Detection (BOD)**
- **Calibration of the RC Oscillator**
- **Reset Source Register**
- **Band-gap Reference on the Comparator**
- **EEPROM Interrupt**

Also note that the startup times differ slightly between the devices. Please refer to the datasheets for more detailed information.

6.1.3 Using the AT90S2313 Adapter for Emulating the AT90S1200

The AT90S1200 can be defined as a subset of AT90S2313. They have the same pinout, but AT90S1200 does not have the following features:

- UART
- SRAM
- Memory Access Instructions (ld/st/lds/sts/ldd/std/lpm)
- 16-bit Arithmetic Instructions (adiw/sbiw)
- INT1
- Timer/Counter 1 and Input Capture
- Stack Pointer to SRAM (AT90S1200 has a 3 level hardware stack)

Avoiding the use of these features and using only half the program and EEPROM memories allows the AT90S2313 to be used when emulating AT90S1200.

IMPORTANT!

Since the AT90S2313 uses a stack pointer, this has to be initialized. The simplest way is include the following lines at the top of the program code:

(AVR Assembler example)

```
ldi    r16, 0x65      ; Set the stack pointer to point at the address to
                       ; give a 3-level deep stack
out     0x3D, r16
```

The AT90S2313 has no RC oscillator, so this feature found on the AT90S1200 can not be supported.

Since the AT90S2313 features the EEMWE bit for writing data to the EEPROM memory, this must also be added to the AT90S1200 code if the EEPROM is used.

Include the AT90S2313 file when emulating AT90S1200 to get the interrupts placed on the right locations, see Section 6.1.1.

6.2 AVR Emulator Chip Errata

Latest errata is found on the Atmel web site: www.atmel.com.

| | | |
|--------------|-----------------------------|--|
| 6.3 | Troubleshooting | If you experience problems when installing AVR Studio, connecting the emulator or configuring the emulator, first of all use the checklists in the previous sections to confirm that you have done installation and the setup of the emulator correctly. |
| 6.3.1 | Feedback and Support | <p>To get correct answers to your problems, please include the following details in your request.</p> <p>ICE 200/AVR Studio:</p> <ul style="list-style-type: none">■ Details of which release of ICE 200 you are using■ Details of the platform on which you are running (OS, amount of memory, etc.)■ A small stand-alone sample of code which reproduces the problem■ A clear explanation of what you expected to happen, and what actually happened■ The commands or menu selections you used■ Sample output illustrating the problem■ The information shown in the About dialog box in AVR Studio (version numbers)■ The emulated device <p>Documentation:</p> <ul style="list-style-type: none">■ The manual title and revision■ The page number(s)■ A concise explanation of the problem <p>General suggestions for additions and improvements are also welcome</p> |
| 6.4 | Contact Information | For technical support, please contact your distributor, Atmel sales representative or local Atmel sales office. Atmel sales offices and distributors are listed in the AVR Databook and on the Atmel web site: www.atmel.com . |



Atmel Headquarters

Corporate Headquarters

2325 Orchard Parkway
San Jose, CA 95131
TEL (408) 441-0311
FAX (408) 487-2600

Europe

Atmel U.K., Ltd.
Coliseum Business Centre
Riverside Way
Camberley, Surrey GU15 3YL
England
TEL (44) 1276-686-677
FAX (44) 1276-686-697

Asia

Atmel Asia, Ltd.
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

Japan

Atmel Japan K.K.
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Operations

Atmel Colorado Springs

1150 E. Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL (719) 576-3300
FAX (719) 540-1759

Atmel Rousset

Zone Industrielle
13106 Rousset Cedex
France
TEL (33) 4-4253-6000
FAX (33) 4-4253-6001

Fax-on-Demand

North America:

1-(800) 292-8635

International:

1-(408) 441-0732

e-mail

literature@atmel.com

Web Site

<http://www.atmel.com>

BBS

1-(408) 436-4309

© Atmel Corporation 1999.

Atmel Corporation makes no warranty for the use of its products, other than those expressly contained in the Company's standard warranty which is detailed in Atmel's Terms and Conditions located on the Company's web site. The Company assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. No licenses to patents or other intellectual property of Atmel are granted by the Company in connection with the sale of Atmel products, expressly or by implication. Atmel's products are not authorized for use as critical components in life support devices or systems.

Microsoft, Windows and Windows NT are trademarks of Microsoft Corporation. Pentium is a trademark of Intel Corporation. Acrobat Reader is trademark of Adobe Systems Incorporated.

All other marks bearing ® and/or ™ are registered trademarks and trademarks of Atmel Corporation.

Terms and product names in this document may be trademarks of others.



Printed on recycled paper.

1413A-07/99/5M