IN-CIRCUIT DEBUG (ICD)
USER GUIDE

TABLE OF CONTENTS

TABLE OF FIGURES

1. Introduction

Included with WDC's Development System is the capability to perform In-Circuit Debug (ICD). ICD is used with WDC's microprocessors, Developer Boards (DB) and microprocessor IP.

To interface the SDS Debugger to the host PC, the DB's have a Monitor burned into the onboard EPROM and the physical interface to the PC is through the parallel printer port. The Monitor configures the W65C22S VIA on the DB to interface to the printer port, referred to as the Debug Port.

The WDC Debugger works with both our simulator tool and our hardware (Developer Boards). When the Debugger is used with the Developer Board, it is thought of as "in-circuit debugging". WDC felt that users of ICD would enjoy many of the ICE features without the expense of an ICE system. Also, since ICD is built into the target, field service updates and troubleshooting is made easy through the Debug Port. ICEing a system in the field is difficult if not impossible due to system configuration differences between the development system and the production system. With ICD, the development system and the production system can both have ICD. This can be accomplished using the ICD and a laptop.

All releases of our SDS support ICD. We have recently improved the features of the ICD to include a hardware breakpoint. This breakpoint requires a version of the Xilinx CPLD and a new revision of the Debugger executable (Wdcdb.exe). The user that has the ability to reconfigure the CPLD can implement this change from WDC supplied Verilog RTL code. If the user can't implement the CPLD change, WDC could supply a CPLD with the hardware breakpoint already implemented.

The hardware breakpoint can be set for any address location within the microprocessor address space and can be further specified to break on a READ or WRITE with matching or unmatching data comparison. These features provide powerful ICD capability when combined with the SDS software breakpoints.

For those users who want to have hardware trace for hardware bugs, a logic analyzer is recommended. Once the system is in production, hardware bugs should not cause software development problems; therefore, ICD is becoming an important development tool.

When reusing our microprocessor IP, WDC recommends that our licensees consider designing ICD features (Debug Port, monitor, and supporting register file) right into their ASIC. Embedding ICD into an ASIC, microcontroller, or SoC would eliminate a special bonded chip just for ICE and the ICE system cost. Our developer boards can be thought of as "reference designs" or "evaluation units" for implementing ICD in microcontrollers, ASICs or SoCs that use our microprocessor or microcontroller IP library.

For those users that have an existing design or don't want to design ICD into their ASIC, our developer boards can be plugged into a target system through an Interposer, thus resembling an ICE. The developer board provides the Debug Port, monitor and support files. The Interposer is a 44 pin PLCC socket with extended pins. If the target is not in the 44pin PLCC package, a conversion board will need to be made to interface the developer board to the target system. For more information on using the developer board as an ICE, please refer to section 5 of this manual: ***WDC Developer Board In-Circuit-Debug Considerations When Used as an In-Circuit-Emulator***.

2. Debug Port

When designing an ASIC System-on-a-Chip (SoC) IC, a Debug Port is recommended. The Debug Port has 8 data IO pins and 2 handshake pins. In addition, the ICD debug interface uses NMI of the processor. The host PC manually invokes the Monitor when pushing the ESC key on the host PC keyboard by pulling NMI low on the debug target processor. These two NMI source interrupts are controlled by the ICD circuitry. The SDS debugger determines the source by reading the ICDCTRL[7] Status bit, a 1 indicates that the breakpoint was reached. The SDS Debugger for future breakpoint status must clear this bit by writing a zero to it.

One output, BRKPTB, from the ICD circuitry is provided to trigger a logic analyzer trace. BRKPTB is located on the Memory Bus, J3 pin 44 CPLD PLD19, connector of the DB's. BRKPTB is pulsed low when the breakpoint is reached and can be used to trigger a logic analyzer to begin tracing any signal connected to the logic analyzer.

3. The ICD Registers

ICD registers are addressed reference to an ICD base address. The ICD_BASE address for the W65c02DB and W65c816DB is $(00)00D0, for the W65c122DB and W65c134DB it is $(00)0100 and for the W65c265DB it is $(00)DF10.

Address	RTL Label	Description
0	BRKREG0	Address byte 0 (bits 0-7)
1	BRKREG1	Address byte 1 (bits 8-15)
2	BRKREG2	Address byte 2 (bits 16-23)
3	BRKREG3	Address byte 3 (bits 24-31) Reserved
4	DATREG0	Data Compare Value byte 0 (bits 0-7)
5	DATREG1	Data Compare Value byte 1 (bits 8-15) Reserved
6	DATREG2	Data Compare Value byte 2 (bits 16-23) Reserved
7	DATREG3	Data Compare Value byte 3 (bits 24-31) Reserved
8	Reserved	
9	Reserved	
A	Reserved	
B	Reserved	
C	Reserved	
D	Reserved	
E	Reserved	
F	ICDCTRL	ICD Control Register

4. ICDCTRL Register Bit Definitions

Bit	RTL Label	Description	0	1
0	BRKEN	Breakpoint Enable	Disable	Enable
1	RWSEL	Read or Write Select	Write	Read
2	DATAEN	Data Compare Enable	Disable	Enable
3	MATCH	Data Match Select	Doesn't Match	Match
4	Reserved			
5	Reserved			
6	Reserved			
7	BREAK	Breakpoint Status	No Breakpoint	Breakpoint Match

Note: In order to reset the Breakpoint Status, you need to write a zero to it to clear.

5. WDC Developer Board In-Circuit Debug Considerations When Used as an In-Circuit-Emulator

The following are pin function considerations when using the ICD.

ABORTB (W65C816S)

The ABORTB pin is defined as a CPLD output pin and is held to a "1" on the DB. When the DB is used to debug a target system, the CPLD pin should be defined as an input pin, allowing the target to control the state of ABORTB.

Address Bus, Data Bus, RWB, VDD and VSS

The address bus, data bus, RWB, VDD and VSS should not require any changes. It is recommended that the ICD board receive power from the target, through the CPU socket pins, to prevent power supply sequence problems.

BE

BE is currently held to a "1" and would have to be defined by the target logic and this could be done in the RTL code by changing BE to a CPLD input rather then an output as it currently exists. If the target just holds BE at a "1" then no changes are required.

IRQ

The DB VIA IRQ is controlled by the DB CPLD and the target IRQ (TIRQ) is connected to PLD50 (pin 17 of J2 on the W65c02DB, W65c134DB and W65c265DB, and pin 7 of J2 on the W65c816DB) to provide CPLD control. The interposer IRQ pin can be removed so that TIRQ is disconnected from the DB CPU socket. TIRQ can now be reconnected to PLD50 using a jumper wire. The CPLD will now logically AND the TIRQ with the DB VIA IRQ logic so that a TIRQ active low will be recognized.

NMI

There are three possible sources for an NMI interrupt that must be determined by the debugger for proper operation. One source for an NMI interrupt is the PC ESC key (NMIPIN). The ESC key is used to manually interrupt the system from the PC host. Another source of an NMI interrupt is the ICD hardware breakpoint logic. When a breakpoint is reached, NMI is pulled low for one cycle, invoking the debug monitor. A third source for an NMI interrupt is the target NMI (TNMI) which can be connected to PLD55 (pin 15 of J2 on the W65c02DB, W65c134DB and W65c265DB, and pin 5 of J2 on the W65c816DB). The interposer pin can be removed to disconnect the TNMI from the DB CPU socket. A short pin can now be placed in the target where the interposer pin was removed and a jumper wire may be used to reconnect TNMI to PLD55. The CPLD will now logically AND TNMI into the DB NMI logic so that a TNMI active low will be recognize.

The debugger determines if the breakpoint logic caused the NMI by reading ICDCTRL bit 7 (BREAK). If BREAK is set the hardware breakpoint caused the NMI. The TNMI logic must provide information to the debugger to determine if the TNMI was active.

PHI2

When the target PHI2 clock source is to be used, the oscillator chip is removed so the target PHI2 is used.

RDY

The DB has a wired-OR pull-up resistor on the RDY pin that is gated with RDYOUT so that when RDY is pulled low, internally, by the WAI instruction, the current in the pull-up resistor is eliminated. The DB R2 should be removed when the DB is used as an ICE system. If R2 is not removed the system may work properly because the combination of R2 (3K) in parallel with the target RDY pull-up resistor can still be pulled low.

RESB

When the target RESB is to be used, JP2 is removed.

SRAM and EPROM

The SRAM and EPROM memory map locations will have to consider the target memory map and any conflicts resolved by reconfiguring the CPLD. This may require that the monitor resides in the target space or the target memory removed and the target use the DB memory.

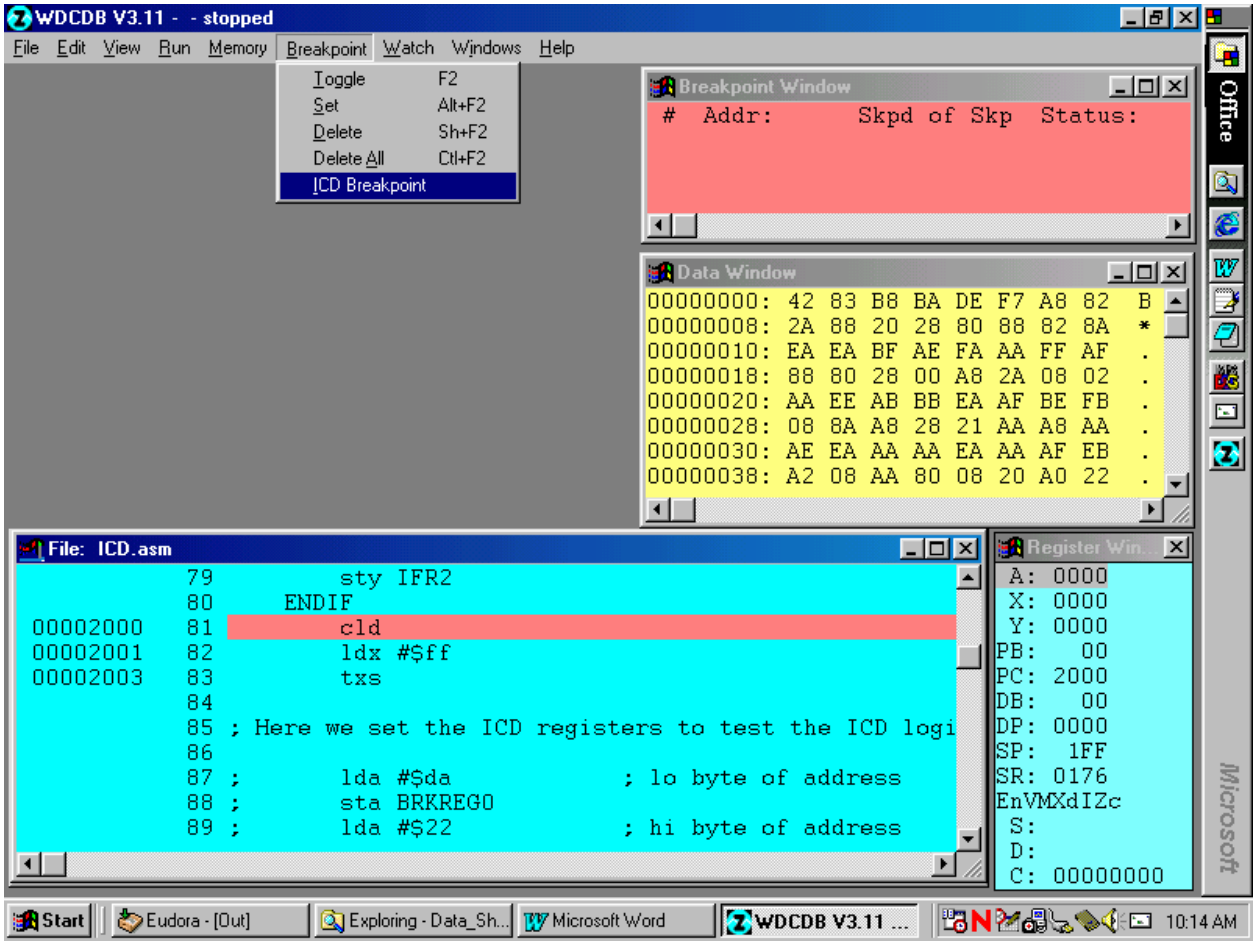

Figure 1. Software Development System (SDS) ICD hardware breakpoint control window selection.

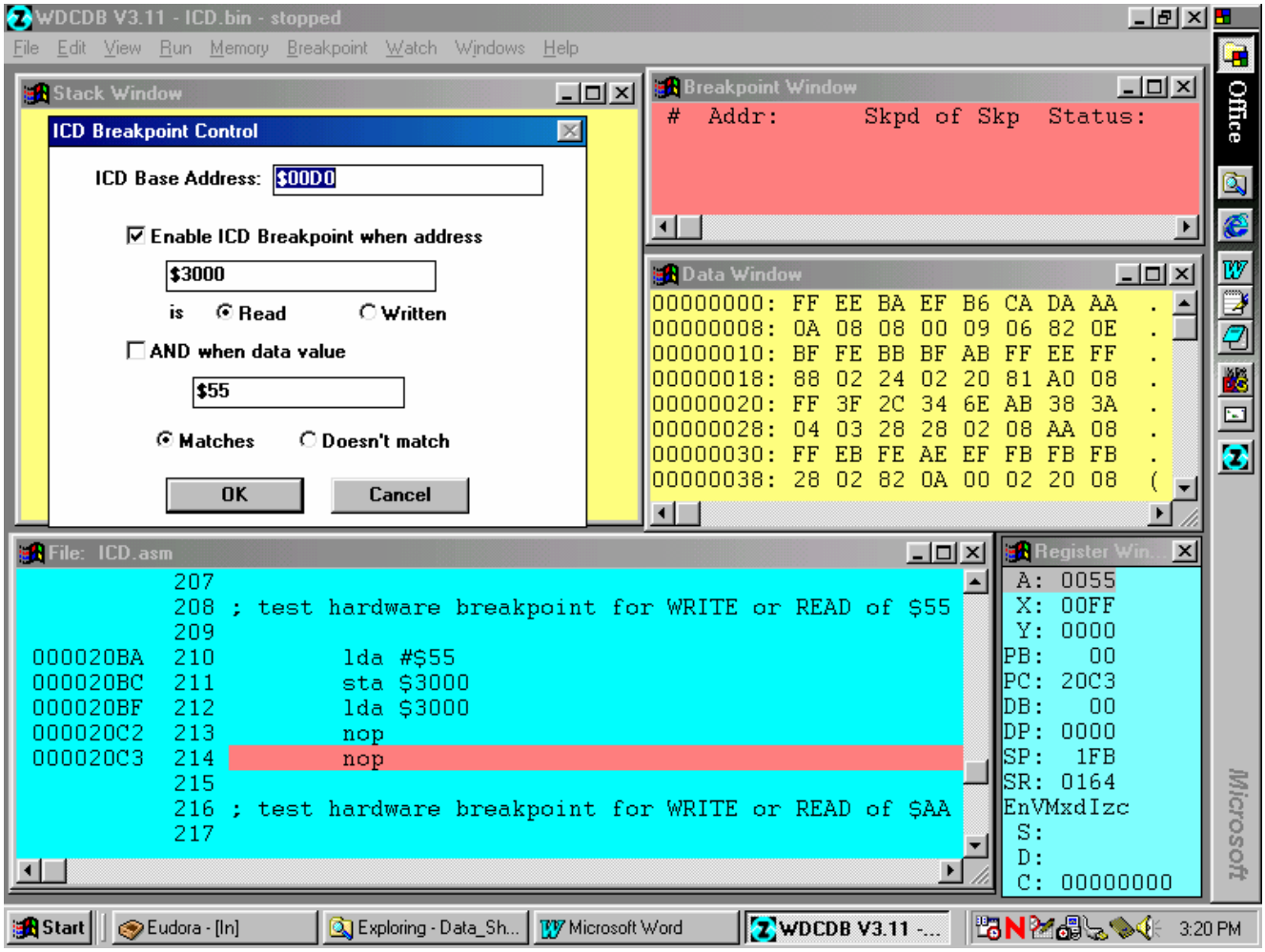

Figure 2. Breakpoint conditions selected for a read from location $3000.

Note that the debugger highlights the second instruction after the breakpoint due to the NMI interrupt response cannot occur immediately on the next instruction. Also, the breakpoint logic for the W65c816S **does not have** the data compare function implemented due to lack of CPLD gates (therefore the box next to AND is not selected). Please make sure to uncheck the box for AND when data value when using the W65C816DB. The data compare for matching or non-matching is implemented for W65c02S, W65c134S and W65c265S ICD's.

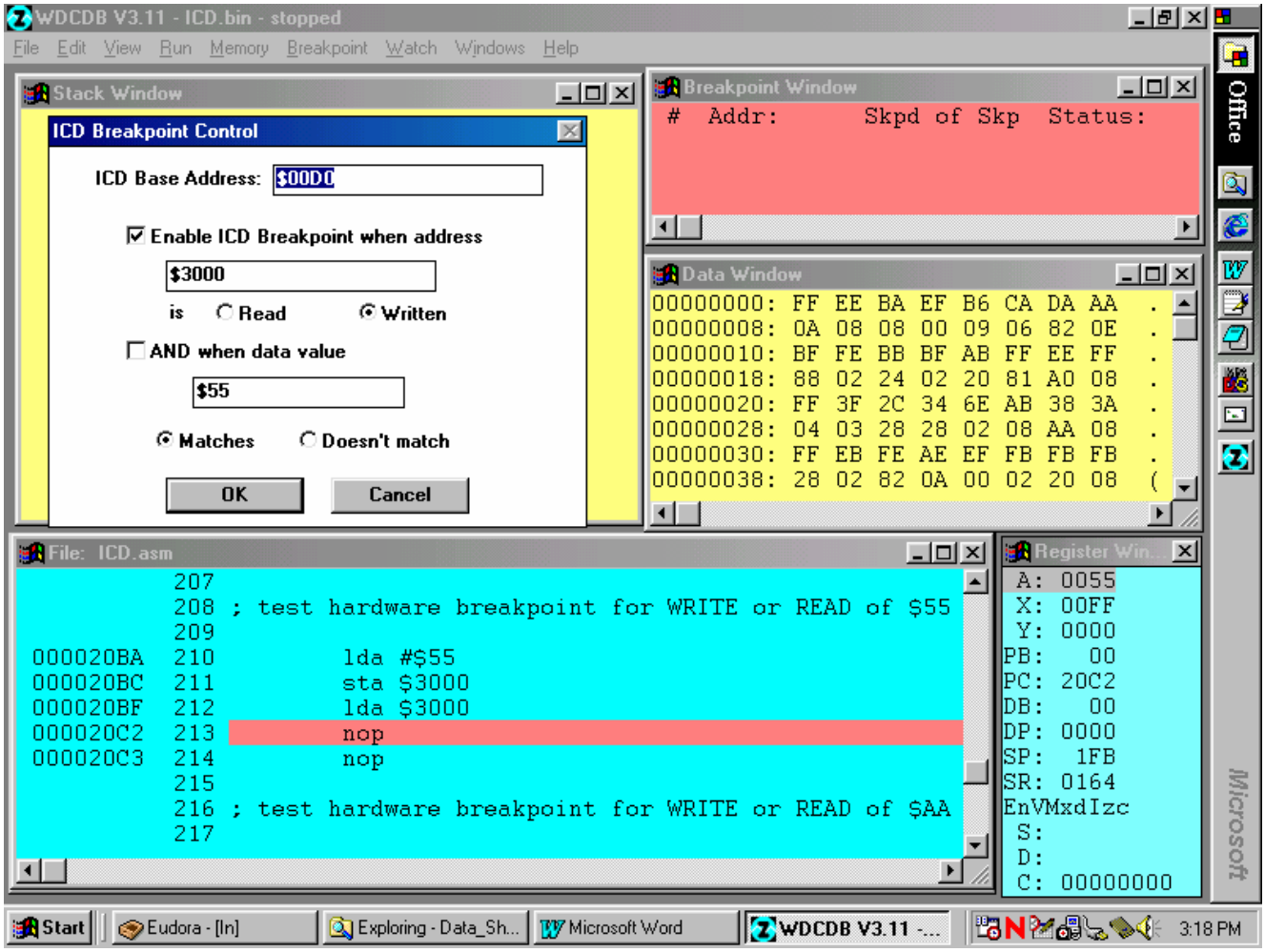

Figure 3. Breakpoint control window with breakpoint conditions selected for a write to location $3000.

Note that the debugger highlights the second instruction after the breakpoint due to the NMI interrupt response cannot occur immediately on the next instruction. Also, the breakpoint logic for the W65c816S does not have the data compare function implemented due to lack of CPLD gates (therefore the box next to AND is not selected). The data compare for matching or non-matching is implemented for W65c02S, W65c134S and W65c265S ICD's..

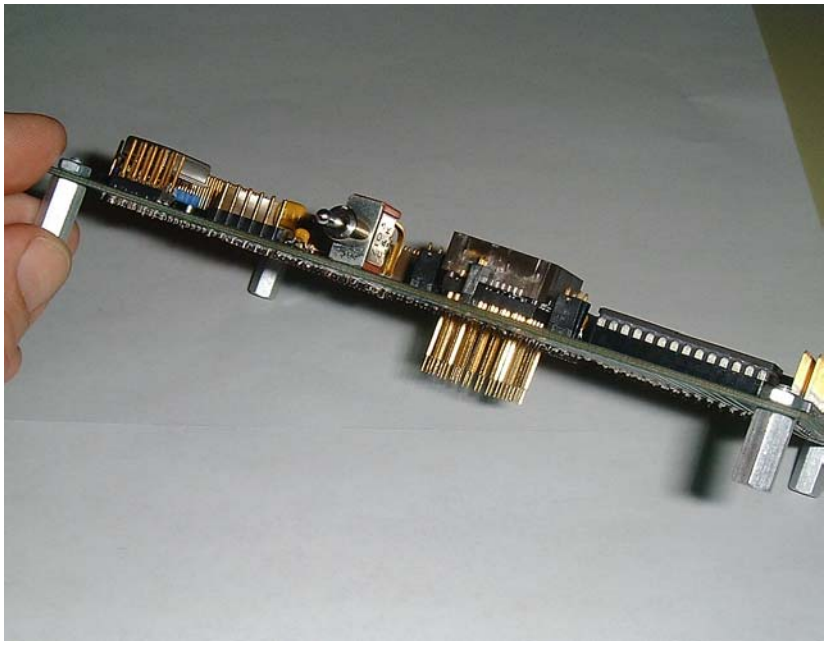

Figure 4. Developer Board with ICD pins that plug into target.

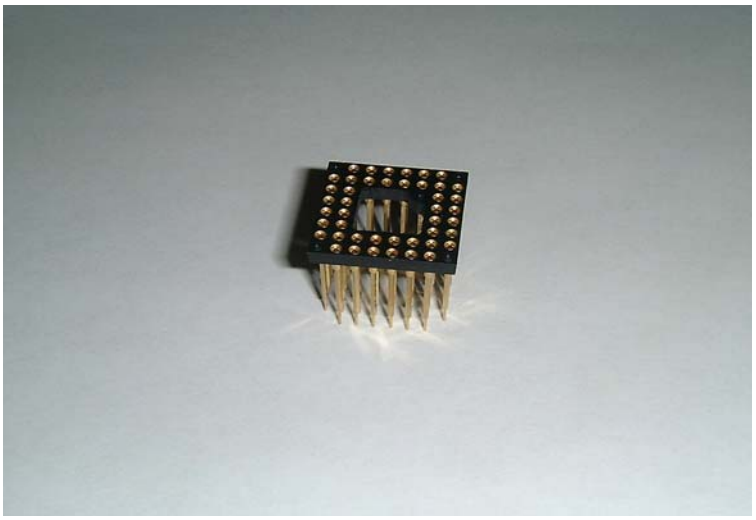

Figure 5. The interposer that can be used to extend the ICD pins and connect the ICD board to the target.

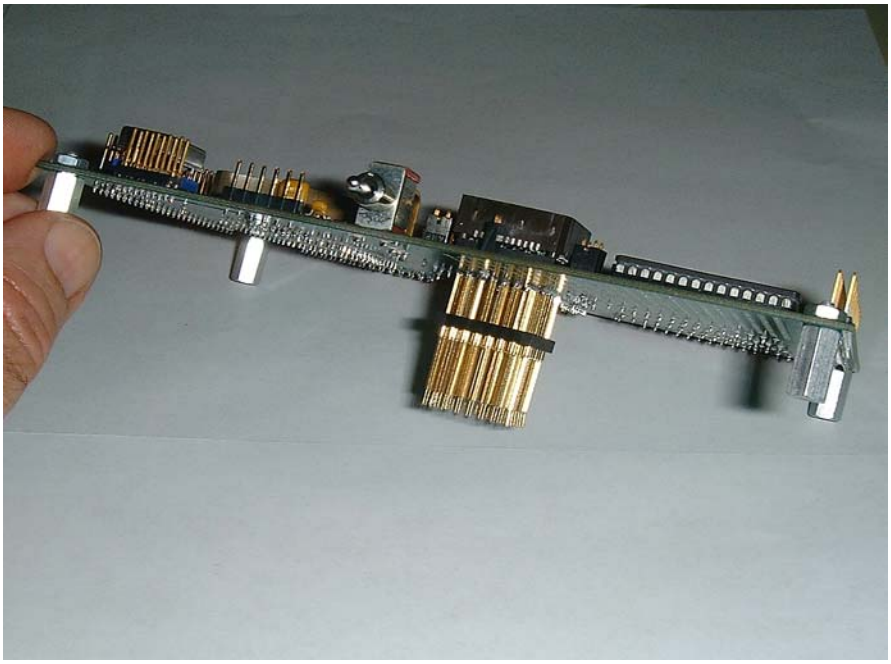

Figure 6. ICD board with Interposer used to connect the ICD board to the target.

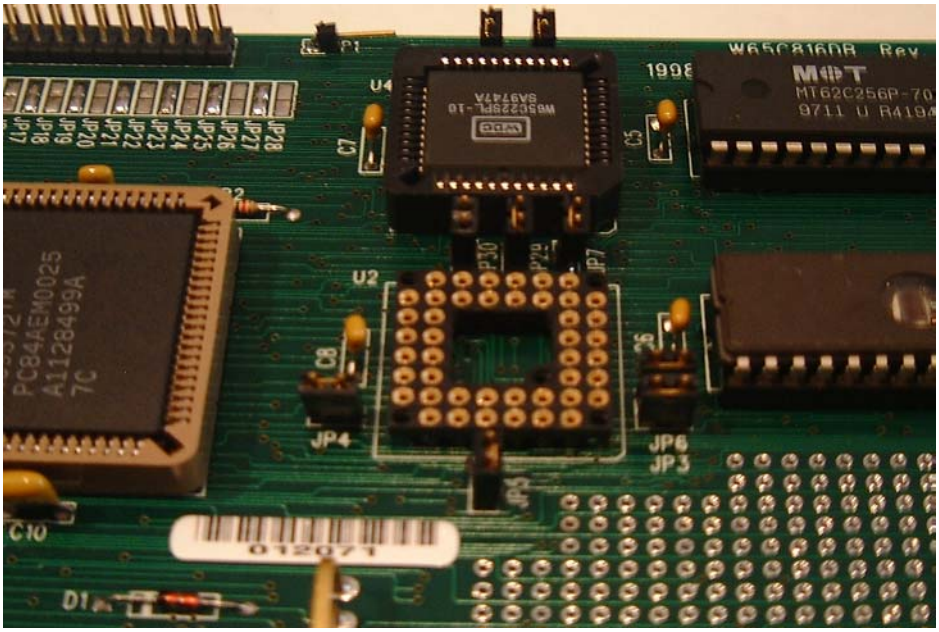

Figure 7. The target's 44 pin PLCC socket used to connect the ICD board to the target.

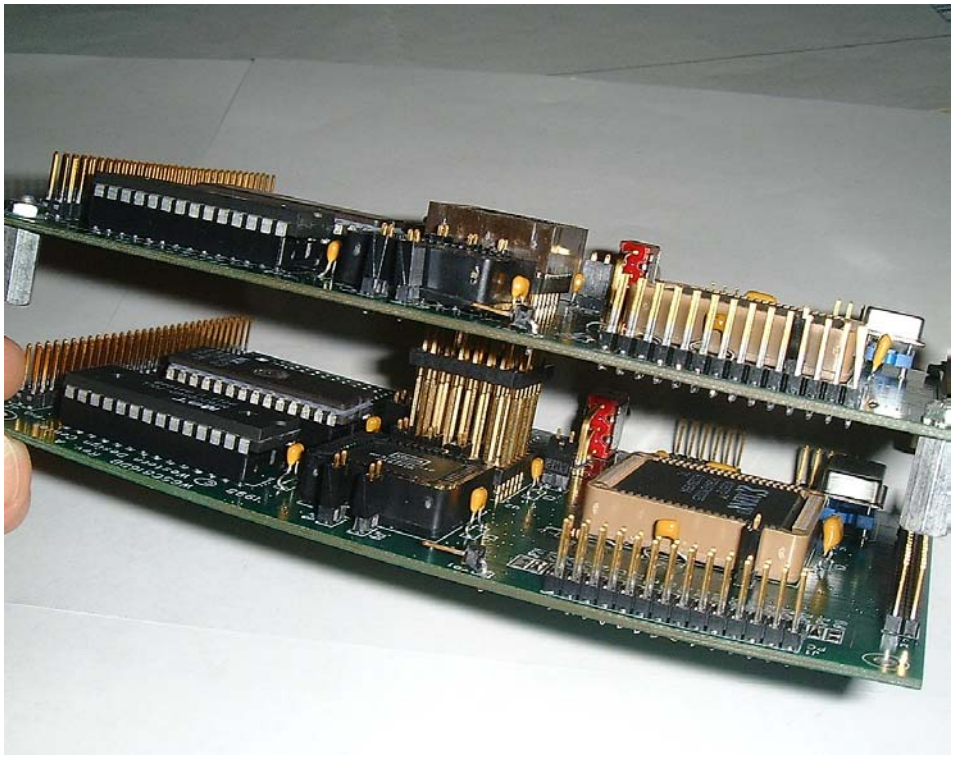

Figure 8. ICD board with Interposer connection to the target

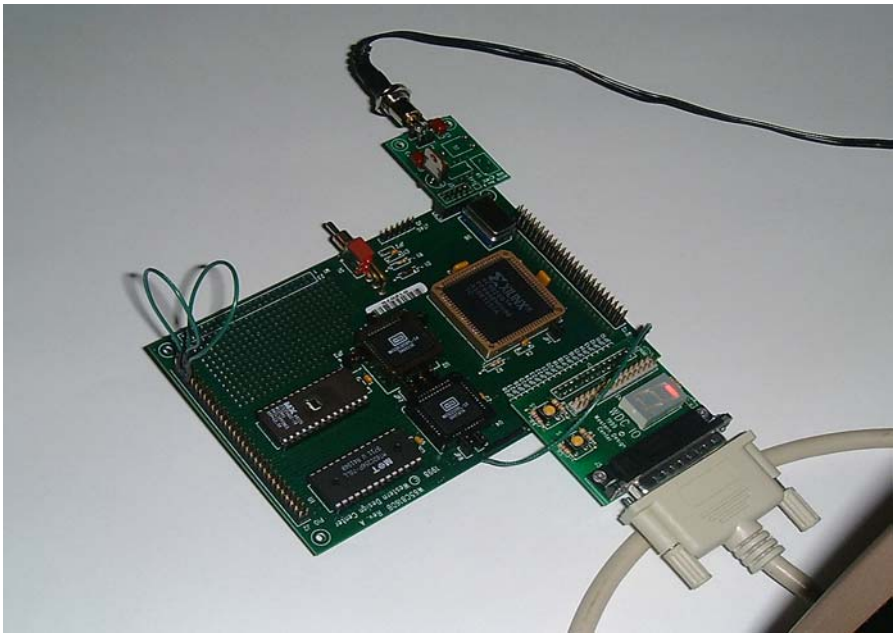

Figure 9. ICD system setup with TIRQ and TNMI jumpers connected to VDD.

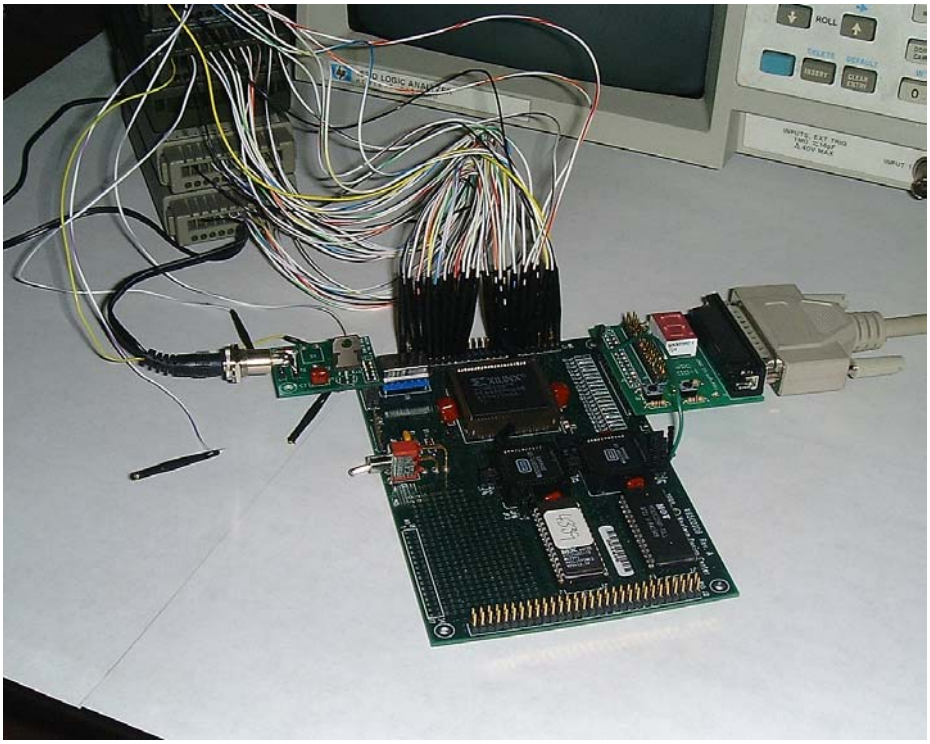

Figure 10. ICD board with HP logic analyzer attached to the memory bus.