URCOM

User's Manual

V1.1

TechTools
PO BOX 462101 Garland, TX 75046
(972) 272-9392 FAX: (972) 494-5814
www.tech-tools.com support@tech-tools.com

Table of Contents

INTRODUCTION

URCOM is a very unique UniROM option. It provides a bi-directional communications path between the TARGET and the HOST PC without using TARGET WRITE cycles. This capability allows a target to communicate through shared memory, even if it is not capable of writing to the memory socket being emulated by UniROM.

INSTALLATION

URCOM is an expansion option for UniROM. It is plugged into a set of connectors on UniROM's base board. The following steps outline the installation procedure:

Verify *Uni*ROM's Firmware Version

Before beginning the installation, verify that your current FIRMWARE revision is 1.52 or newer. To verify your Firmware revision, follow these steps:

Connect to *Uni*ROM as usual with URTerm.exe.
Reset *Uni*ROM
Press the ENTER key until *Uni*ROM responds.
*Uni*ROM will display a sign-on message and its firmware revision.

If your firmware is older than 1.52 and your order did not include a firmware upgrade, call out tech-support department for additional information.

PLEASE observe standard anti-static procedures

*Uni*ROM and URCOM are electronic devices that require basic anti-static precautions while handling. They are no more OR LESS susceptible to damage due to electrostatic discharge than other electronic assemblies and should be handled only at an anti-static equipped work-station. If your facilities are not equipped to provide these basic precautions, we would be happy to do the installation for you at no charge. Call our Tech-support department for an RMA number and return instructions.

Remove *Uni*ROM's case

Remove the screws from the bottom of *Uni*ROM with a #1 Phillips screwdriver. Remove the case TOP and BOTTOM.

Place *Uni*ROM's circuit board assembly on an anti-static mat or other flat, grounded, conductive surface.

*Uni*ROM is an electronic device that can be damaged by static electricity. It should be handled with the appropriate precautions.

Remove *Uni*ROM's rear end-plate

The rear end-plate is held in place by the case top and bottom; no mounting hardware is used. To remove the end-plate, move all cable latches to their "latched" positions and slide the end-plate off.

Remove *Uni*ROM's memory board(s)

*Uni*ROM has one or two memory architecture boards plugged into a base board. The memory board(s) must be removed to gain access to the URCOM expansion connectors. If your *Uni*ROM has two memory boards, there is no need to separate them from each other. They may be removed as single unit.

The memory architecture board connectors have a very firm grip. You will need to "work loose" all four corners before the board(s) can be removed. This can take some time, but resist the urge to "pry" the boards apart. This procedure could very easily damage other components or board traces.

BE VERY CAREFUL NOT TO DAMAGE THE BASE BOARD OR MEMORY BOARDS!
Our warranty WILL NOT cover such damages.

If you are uncomfortable with this procedure, we would be happy to do the installation for you at NO CHARGE. Simply call out Tech-support department for an RMA number and return instructions.

Install URCOM board

Insert URCOM into the expansion connectors found directly below the memory board(s). NOTE that URCOM is installed with its components facing down.

Check carefully that all pins are aligned properly with the sockets.

It will take considerable force to install URCOM, so verify that the *Uni*ROM baseboard has been placed on a firm, flat surface (NOT on the case bottom). Press URCOM into its socket with firm, even pressure.

Visually inspect the installation to verify that the connectors are completely seated and that NO PINS were bent. When properly inserted, the pins' plastic body should touch the sockets' plastic body and the URCOM components should almost touch the *Uni*ROM base board.

Upgrade the firmware (if needed)

If you are upgrading your firmware, use a PLCC removal tool to remove the old ROM at U5. This is a socketed 32 pin PLCC device. The socket can be damaged if a screw-driver or other substitute is used in place of a PLCC removal tool.

Rotate the replacement Firmware ROM so that its notched corner aligns with the socket's notched corner. The notch is used to ensure that the ROM is not inserted backwards, but one could forced the part in backwards with enough effort.

Square the ROM in the socket so that the socket's small ribs fit BETWEEN the ROM's pins.

When the part is aligned properly, insert it into the socket with firm, even pressure. When properly seated, the top of the ROM should be flush with, or slightly below the top of the socket.

Re-install the memory board(s)

Re-insert the memory boards into the memory expansion sockets, being careful to orient the board such that the target connector is pointing towards the rear.

INSERTING THE MEMORY BOARD BACKWARDS WILL RESULT IN DAMAGE TO THE BOARD WHEN POWER IS APPLIED!

Seat the board(s) firmly into the connectors. You are inserting 80 pins into full contact sockets, so this will take considerable force.

Re-install the end-plate

Slip the end-plate over the target connectors.

Re-install *Uni*ROM in its case

Place the circuit board assembly back on the case BOTTOM, placing both end-plates into the molded grooves.

Place the case TOP into place. Verify that the case top seats fully and that the end-plates are seated into the molded groves in the TOP and BOTTOM pieces.

Turn *Uni*ROM over and re-insert the screws.

URCOM CONFIGURATION

URCOM is configured through the SETUP\HARDWARE UART menu. Use URTERM to set URCOM's address and optionally enable its interrupt. The address can be any 16 byte aligned address within the memory space emulated by *Uni*ROM. A 16 byte aligned address will always end in a '0' in HEX notation (0xfff0, 0x3560, 0xd0000 ...).

Use SETUP\CONSOLE menu to connect a HOST port to the URCOM device during console operations.

A virtual connection will now be made between the HOST port and the URCOM option board whenever "CONSOLE" is selected from the main menu, or whenever URLOAD is run with "CONSOLE=ON" specified in its INI file.

SPECIAL NOTES for DUAL (16bit) *Uni*ROMs

IF you are using a 16bit *Uni*ROM in "TWO INDEPENDENT DEVICES" organization, URCOM MUST be mapped into the lower emulation board.

If you are using a 16bit *Uni*ROM in "INTERLEAVED' organization, the URCOM board will occupy 32 bytes of memory.

COMMUNICATING THROUGH URCOM

This section describes how the target uses URCOM to send and receive data. Each operation is described below.

RECEIVE

The target receives data from URCOM very much like it would from a standard UART. A STATUS register holds a RECEIVE FULL flag that indicates when the RECEIVE BUFFER holds a byte of data. When the target reads the data from the RECEIVE BUFFER, URCOM automatically clears the flag.

IF the URCOM interrupts were enabled in the SETUP\HARDWARE UART menu, *Uni*ROM will generate an interrupt each time it places data in the RECEIVE BUFFER. The interrupt is cleared when the target reads the status register.

TRANSMIT

This is where URCOM differs from a typical memory-mapped UART. The target "TRANSMITS" by doing a sequence of 3 reads from specific registers within URCOM. Special read sequences are interpreted by URCOM as a request to send specific data. The data to be sent is decoded from the specific registers accessed during the read sequence. When URCOM detects one of these sequences, it places the decoded data into the TRANSMIT BUFFER and sets the TRANSMIT BUFFER FULL flag. When *Uni*ROM transfers this data to a HOST port, the TRANSMIT FULL flag is cleared.

The target determines which registers to read, based on the data it wishes to send. The first read encodes the first 3 bits (D0-2)of the data, the second read encodes the next three bits (D3-5) and the last read encodes the last 2 bits and sets the FULL flag. The data is encoded by reading a specific register, as determined by the data bits. The proper register is determined by ADDING the VALUE of the data bits to the base address of the URCOM as follows:

1st READ address = URCOM_ADDRESS + [the value of (D0-2)].
This will access one of URCOM's registers between 0 and 7.
2nd READ address = URCOM_ADDRESS + [the value of (D3-5)].
This will access one of URCOM's registers between 0 and 7.
3rd READ address = URCOM_ADDRESS + [the value of (D6-7)] + 8
This will access one of URCOM's registers between 8 and 11.

The extra '8' added to the last read tells URCOM that these are the final two bits and the end of this "TRANSMISSION". When URCOM detects a read to one of these four bytes(8-11), it will set the TRANSMIT FULL flag.

Once *Uni*ROM sees the TRANSMIT BUFFER FULL flag, it will transfer the byte to the selected HOST port. URCOM will clear the FULL flag as soon as this byte is read by *Uni*ROM.

Code Examples

The following C code fragments should make this concept more clear.

```
#include <stdio.h>

// assume we configured the URCOM address to 0xF000 in the UniROM setup.
#define URCOM_BASE  (unsigned char *)0xF000

// prototypes
void urcom_init(void);
unsigned char urcom_getch(void);
void urcom_putch(unsigned char c);

// global pointers
static unsigned char *tx_base;
static unsigned char *rx_buff;
static unsigned char *status;

//
// A simple example to demonstrate how a TARGET communications
// through URCOM without using WRITE cycles.
//

void main(void)
{  unsigned char c;

   urcom_init();      // initialize URCOM

   while(1)
     { c = urcom_getch();  // wait for a byte from the HOST
       urcom_putch(c);     // ECHO it back to the HOST
     }

}

void urcom_init(void)
{ unsigned char dummy;

  // initialize pointers
  tx_base = URCOM_BASE;
  rx_buff = URCOM_BASE + 0x0C;
  status  = URCOM_BASE + 0x0D;

  // clear any pending RX interrupts
  dummy = *rx_buff;

}
```

```
unsigned char urcom_getch(void)
{
 // wait for character to appear
 while(!(*status & 0x02))
    ;
 return(*rx_buff);
}



void urcom_putch(unsigned char c)
{unsigned char dummy;

 // wait for TX buffer to empty
 while((*status & 0x01))
    ;
 // send data
 dummy = *(tx_base + (c & 0x07));       // encode d0-d2
 c = c >> 3;
 dummy = *(tx_base + (c & 0x07));       // encode d3-d5
 c = c >> 3;
 dummy = *(tx_base + (c & 0x03) + 8);   // encode d6-d7 and set FLAG

}
```

URCOM REGISTER DEFINITIONS

Offset from BASE	TARGET READ
0-7	Encode D0-2(1st RD), D3-5(2nd RD)
8-11	Encode D6-7 & set TXFULL
12	Read RX BUFFER & Clear RXFULL
13	Read STATUS Register & Clear IRQ
14	-
15	-

STATUS REGISTER DEFINITION (TARGET SIDE)
D0: TX FULL FLAG 1-> last byte not received by HOST yet
D1: RX FULL FLAG 1-> Receive data available
D2-7: Unknown

When the HOST DISABLES URCOM:
- the status register is cleared
- the target sees the underlying SRAM instead of URCOM registers.

CONCLUSION

We designed URCOM so that it can be easily integrated in any target with minimal work. This option allows one to add full bi-directional communications to any target without hardware modifications and with minimal software modifications.

If you would like to share your own URCOM routines for your specific target CPU with our other customers, feel free to upload them to our BBS. We would also welcome any comments on your particular applications for this device.

If you have any problems, comments or suggestions, please feel free to contact us through VOICE, FAX or our BBS.

TechTools
PO BOX 462101 Garland, TX 75046
(972) 272-9392 FAX: 494-5814 email: support@tech-tools.com