



XAPP982(v1.0) March 12, 2007

Reference System: OPB IIC Using the ML402 Evaluation Platform

Author: Paul Glover

Summary

This application note describes how to build a reference system for the On-Chip Peripheral Bus Inter IC (OPB IIC) core using the MicroBlaze™ based embedded system in the ML402 Embedded Development Platform. The reference system is Base System Builder (BSB) based. This application note provides information for most users who need to simulate the OPB IIC or use it in MontaVista Linux or use ChipScope.

The Xilinx Microprocessor Debugger (XMD) commands are used for verifying that the OPB IIC core operates correctly. Several software projects illustrate how to configure the OPB IIC core, set up interrupts, and do read and write operations. Some of the software projects interface the OPB IIC core to the MicroChip Technology 2L024B serial EEPROM with an IIC interface.

Included Systems

This application note includes one reference system:

www.xilinx.com/bvdocs/appnotes/xapp982.zip

The project name used in xapp982.zip is ml402_mb_opb_iic.

Required Hardware/Tools

The following tools, cables, peripherals, and licenses must be available and installed:

- Xilinx EDK 8.2.02
- Xilinx ISE 8.2.03
- Xilinx Download Cable (Platform Cable USB or Parallel Cable IV)
- ML402 Board

Introduction

This application note accompanies a reference system built on the ML402 development board. [Figure 1](#) is a block diagram of the reference system.

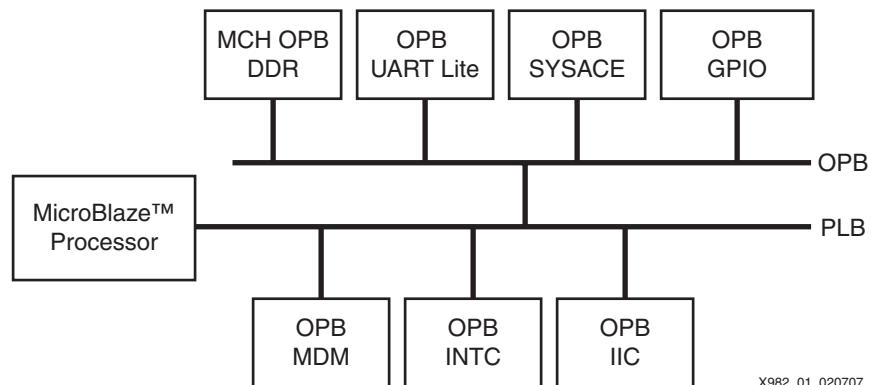


Figure 1: OPB IIC Reference System Block Diagram

© 2007 Xilinx, Inc. All rights reserved. All Xilinx trademarks, registered trademarks, patents, and further disclaimers are as listed at <http://www.xilinx.com/legal.htm>. PowerPC is a trademark of IBM Inc. All other trademarks and registered trademarks are the property of their respective owners. All specifications are subject to change without notice.

NOTICE OF DISCLAIMER: Xilinx is providing this design, code, or information "as is." By providing the design, code, or information as one possible implementation of this feature, application, or standard, Xilinx makes no representation that this implementation is free from any claims of infringement. You are responsible for obtaining any rights you may require for your implementation. Xilinx expressly disclaims any warranty whatsoever with respect to the adequacy of the implementation, including but not limited to any warranties or representations that this implementation is free from claims of infringement and any implied warranties of merchantability or fitness for a particular purpose.

Reference System Specifics

Table 1 provides the address map of the ML402 XC4V5X35.

ML402 XC4VSX35 Address Map

Table 1: ML402 XC4VSX35_ff668 System Address Map

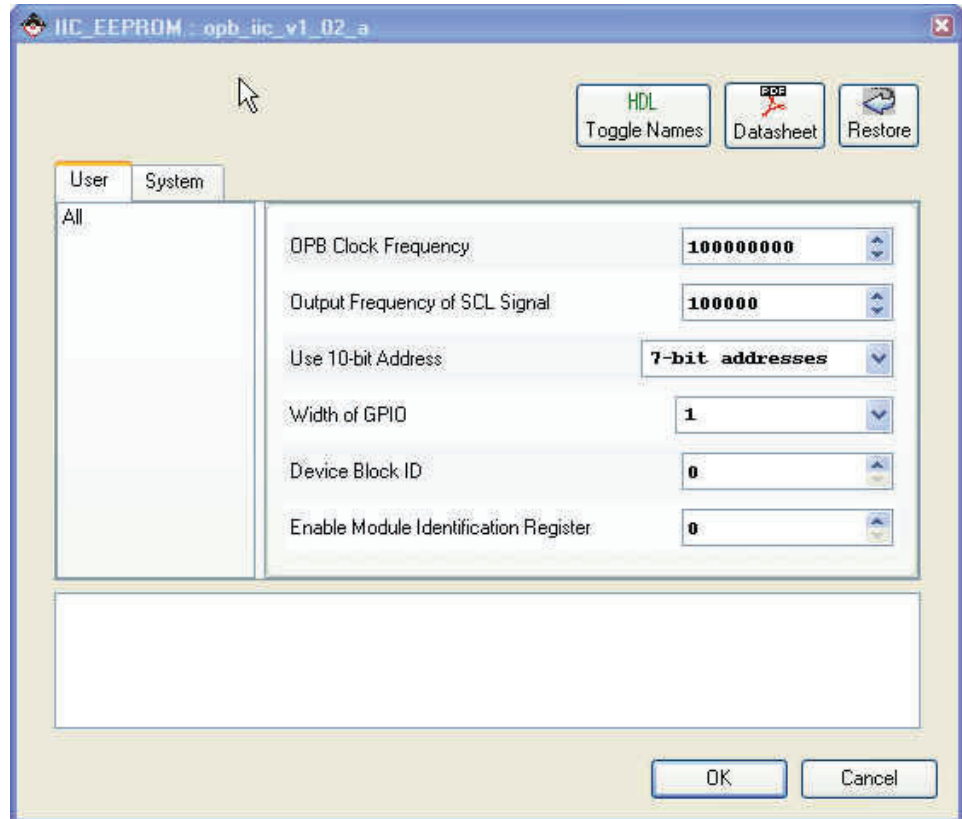
Peripheral	Instance	Base Address	High Address
MCH_OPB_DDR	DDR_SDRAM_64Mx32	0x24000000	0x27FFFFFF
OPB UARLITE	RS232_Uart	0x40600000	0x4060FFFF
OPB SYSACE	SysACE_CompactFlash	0x41800000	0x4180FFFF
OPB GPIO	LEDs_4Bit	0x40000000	0x4000FFFF
OPB MDM	debug_module	0x41400000	0x4040FFFF
OPB INTC	opb_intc_0	0x41200000	0x4120FFFF
OPB IIC	IIC_EEPROM	0x40800000	0x4080FFFF

Figure 2 shows how to specify the values of generics of the OPB IIC in EDK.

There are several parameters on OPB IIC core that control the functionality of the core. Detail information regarding each of these parameters can be found in the OPB IIC data sheet under the Design Parameters section.

To modify these values in XPS, right click on the IIC_EEPROM in the System Assembly View 1 and select **Configure IP...**

Note: The values shown in Figure 2 are the default values.



X982_02_010507

Figure 2: Specifying the Parameter Values for the OPB IIC Core

Microchip 24LC04B

The Microchip Technology 24LC04B-I/ST with 4-KB EEPROM is provided on the ML402 board to store non-volatile data. The EEPROM write protect is tied off on the board to disable its hardware write protect. The IIC bus is extended to the expansion connector to allow additional devices to be added to the IIC bus.

Figure 3 shows IIC Bus devices on the ML402.

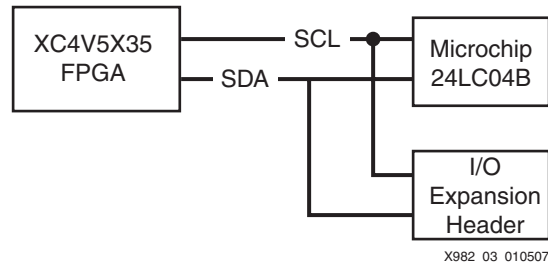


Figure 3: ML402 IIC Bus

The 24LC04B Microchip is organized as two or four blocks of 256 bytes. It has a page write buffer of up to 16 bytes. The 24LC04B operates as an IIC slave. It accepts a control byte which contains control code, block select, and read/write fields. The write transactions are either a byte write or a page write. The 24LC04B supports current address, random, and sequential read operations.

Executing the Reference System using the Pre-Built Bitstream and the Compiled Software Applications

To execute the system using files inside the `m1402_mb_opb_iic/ready_for_download` directory, follow these steps:

1. Change to the `m1402_mb_opb_iic/ready_for_download` directory.
2. Use iMPACT to download the bitstream by using the following:

```
impact -batch xapp982.cmd
```
3. Invoke XMD and connect to the MicroBlaze processor by the following command:

```
xmd -opt xapp982.opt
```
4. Download the executable by the following command

```
dow <path>/executable.elf
```

Executing the Reference System from EDK

To execute the system using EDK, follow these steps:

1. Open `system.xmp` inside EDK.
2. Use **Hardware** → **Generate Bitstream** to generate a bitstream
3. Download the bitstream to the board using **Device Configuration** → **Download Bitstream**.
4. Invoke XMD with Debug Launch XMD.
Download the executable by the following command.

```
dow <path>/executable.elf
```

Verifying the Reference Design with Xilinx Microprocessor Debugger

After downloading the bitstream file, use the following procedure to verify that the ML402 reference design is set up correctly. Invoke XMD and read and write the OPB IIC registers as shown below.

```
mrd 0x40800000 8
```

This will return all zeros.

```
mwr 0x40800100 0xFFFFFFFF
```

This will write 0xFFFFFFFF to the OPC_IIC control register.

```
mrd 0x40800100 1
```

This will read the results back from the OPB_IIC control register. It will return 0000007F.

Software Projects

The three software applications included with the design are described below.

TestApp_Memory: Performs memory tests on the MCH_OPB_DDR memory. This is the application included in the `download.bit` file.

TestApp_Peripheral: Performs basic peripheral tests. In order to run this application the user must load the application into memory via XMD.

iic_eeprom_rw_test: Performs several read and write tests. As shown in [Figure 4](#), the application code contains several definitions which configure the application to match the hardware. To run this application the user must load the application into memory via XMD.

Additionally, there are two defines which control test and debug information provided.

```
/* The following constants map to the XPAR parameters are created in the
 * xparameters.h file. They are defined here such that a user can easily
 * change all the needed parameters in one place. */
#define IIC_BASE_ADDRESS XPAR_IIC_EEPROM_BASEADDR
/* The following constant defines the address of the IIC
 * device on the IIC bus. Note that since the address is only 7 bits, this
 * constant is the address divided by 2.*/
#define EEPROM_ADDRESS 0x50 /* 0xA0 as an 8 bit number */
/* The page size determines how much data should be written at a time.
 * The ML402 board supports a page size of 32 and 16. The write function
 * should be called with this as a maximum byte count.*/
#define PAGE_SIZE 16

/* The Starting address in the IIC EEPROM on which this test is performed */
#define EEPROM_TEST_START_ADDRESS 128

/* This value must be compatible with the hardware
 * Xuint16 for 10 bit addressing
 * Xuint8 for 7 bit addressing */
typedef Xuint8 AddressType;

// Add the define to print debug information
* #define PRINT_DEBUG_INFO
*
//Add the define to print test information
* #define PRINT_TEST_INFO
#define PRINT_TEST_INFO
```

X982_04_010507

Figure 4: Application Code in Software Projects

ML402 Board Information

According to the MicroChip 24L024B data sheet, the ML402 board has a low-level output current (IOL) of 3.0 mA at a VCC of 2.5v. The ML402 boards are shipped in the configuration shown in Figure 5. The board must be modified for this design to work correctly. Replace the 10K Ohm R70 and R71 resistors with 833 Ohm resistors. When testing this design, the the resistors were replaced with a 1K resistor. See Answer Record 24049 for additional information.

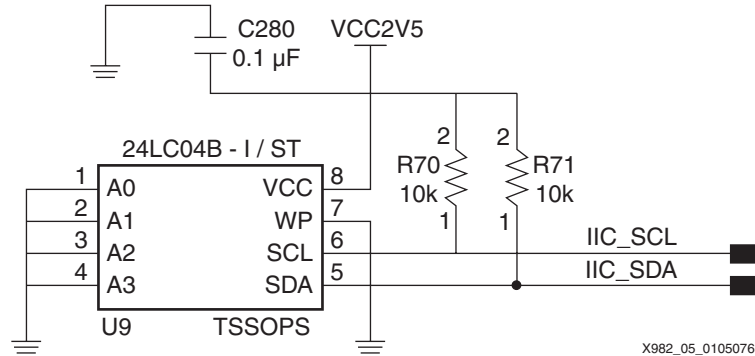


Figure 5: ML40x Schematic for IIC Connections

The resistors are located on the board as shown in Figure 6.

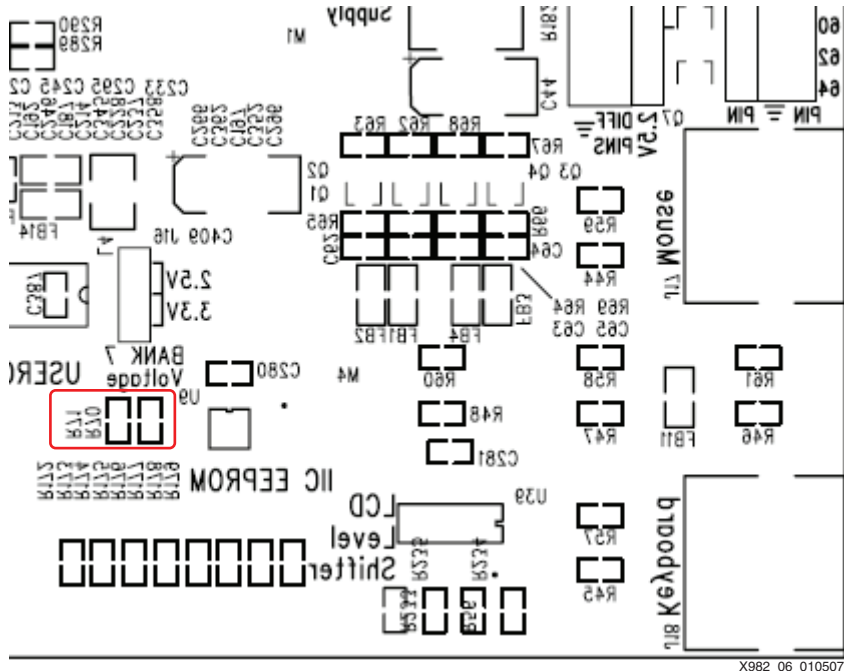


Figure 6: ML40x Resistors

If additional IIC devices are connected to the bus via the expansion header as shown in Figure 7, insert additional pull-up resistors. The resistor values are dependent on the voltage.

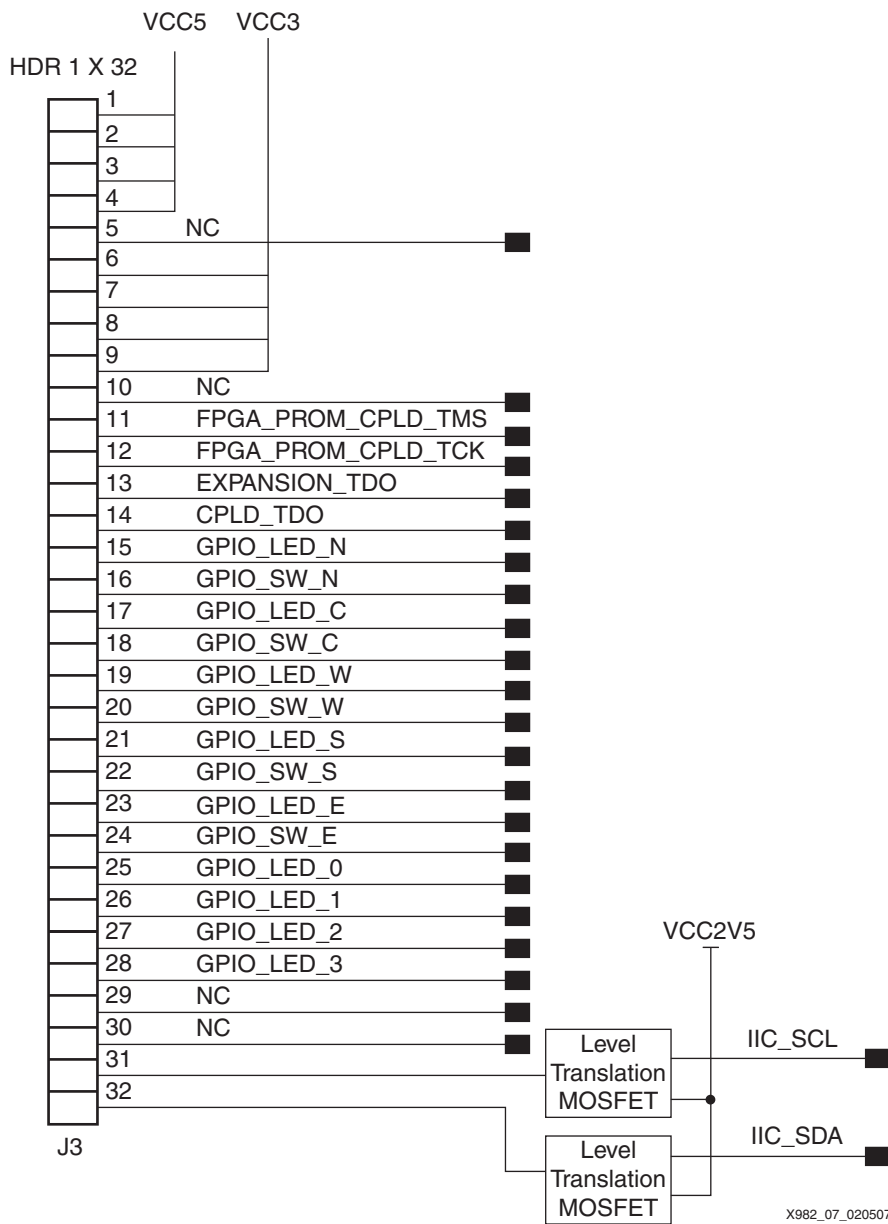


Figure 7: Expansion Header

Figure 8 shows the FPGA pins driving the IIC Bus.

IO_L8N_GC_LC_3_C12	C12	SMA_DIFF_CLK_IN_N	■
IO_L8P_GC_LC_3_C13	C13	SMA_DIFF_CLK_IN_P	■
IO_L7N_GC_LC_3_A17	A17	IIC_SCL	■
IO_L7P_GC_LC_3_B17	B17	IIC_SDA	■
IO_L6N_GC_LC_3_B10	B10	DDR_CLK1_N	■
IO_L6P_GC_LC_3_A10	A10	DDR_CLK1_P	■
IO_L5N_GC_LC_3_A15	A15	DDR_A13	■
IO_L5P_GC_LC_3_A16	A16	DDR_BA1	■
IO_L4N_GC_VREF_LC_3_B12	B12	DDR_BA0	■
IO_L4P_GC_LC_3_B13	B13	DDR_CLK_P	■
IO_L3N_GC_LC_3_C14	C14	MOUSE_DATA	■
IO_L3P_GC_LC_3_C15	C15	PHY_TXCLK	■
IO_L2N_GC_VRP_LC_3_A11	A11	GPIO_LED_2	■
IO_L2P_GC_VRN_LC_3_A12	A12	GPIO_LED_3	■
IO_L1P_GC_CC_LC_3_B14	B14	MOUSE_CLK	■
IO_L1P_GC_CC_LC_3_B15	B15	PHY_RXC_RXCLK	■
FPGA_BANK3			
2.5 VCC0			

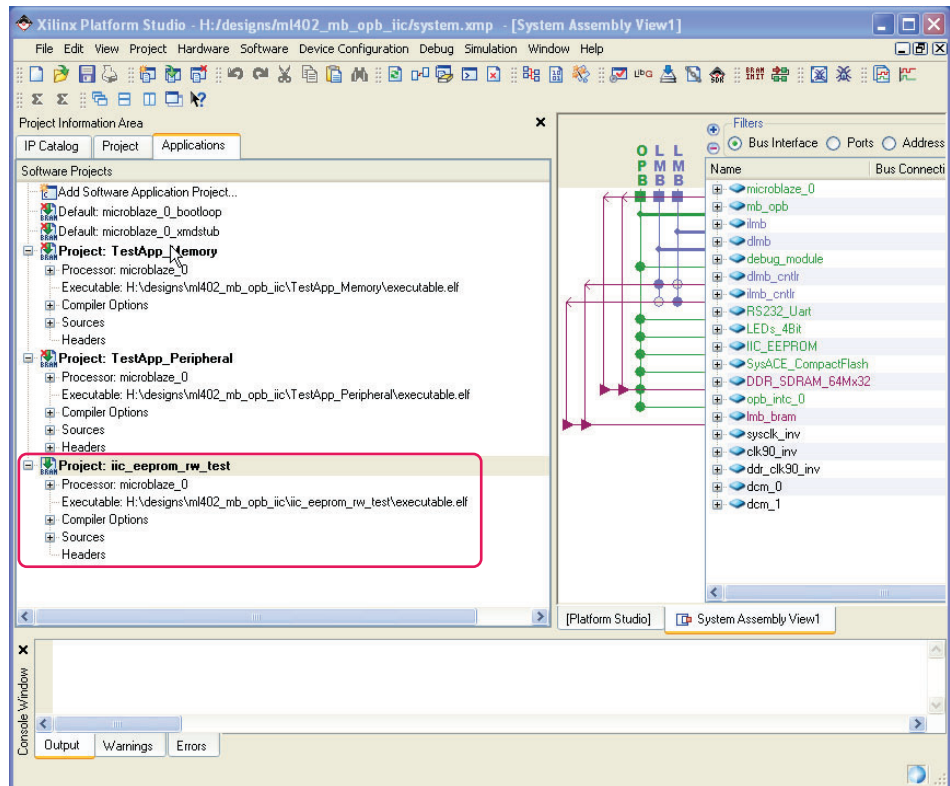
X982_08_030507

Figure 8: FPGA IIC Pins

Running the Applications

In XPS, select **Applications** and **Software Project**.

Figure 9 shows the structure of the iic_eeprom_rw_test software test project. Right click on **Project iic_eeprom_rw_test**, then select **Mark to initialize the BRAM**. Verify that the other projects are not marked to initialize BRAMS.

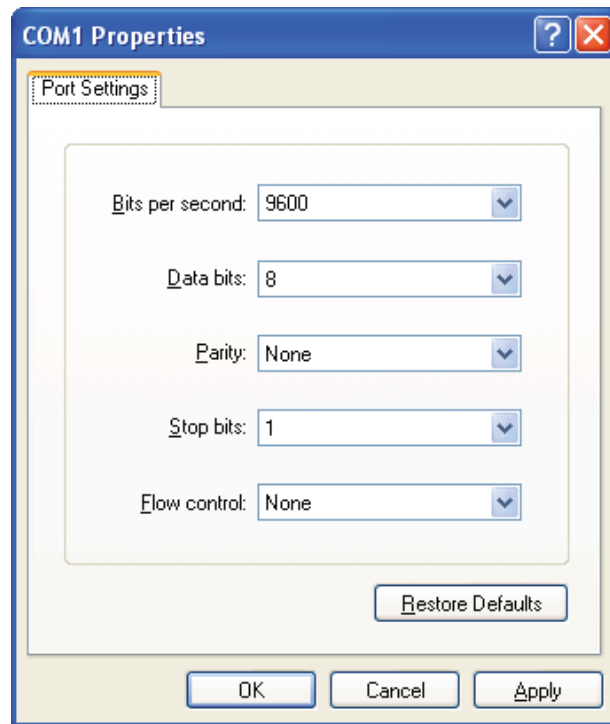


X982_09_020707

Figure 9: Selecting the eeprom Software Project

Select **Project: iic_eeprom_rw_test**, then right click to build the project. If more than one software project is used, make the unused software projects inactive.

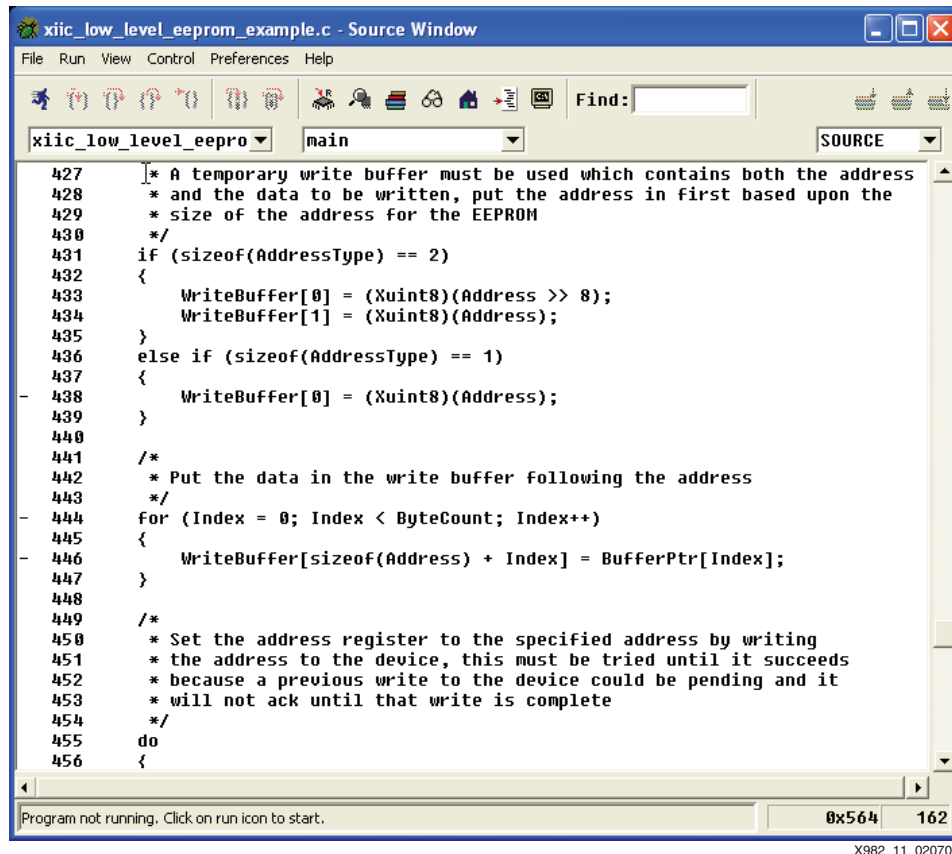
Connect a serial cable to the RS232C port on the ML402 board. Start up a HyperTerminal. Set the Bits per second (baud rate) to **9600**, Data bits to **8**, Parity to **None**, and Flow control to **None**, as shown in [Figure 10](#).



X982_10_010507

Figure 10: HyperTerminal Parameters

From XPS, start **xmd** and enter **rst**. Invoke GDB and select **Run** to start the application as shown in [Figure 11](#). The `xiic_low_level_eeprom_example.c` code written for the ML402 shown in the figure runs without any modifications on this reference system.



```

xiic_low_level_eepro  main  SOURCE
427  /* A temporary write buffer must be used which contains both the address
428  * and the data to be written, put the address in first based upon the
429  * size of the address for the EEPROM
430  */
431  if (sizeof(AddressType) == 2)
432  {
433      WriteBuffer[0] = (Xuint8)(Address >> 8);
434      WriteBuffer[1] = (Xuint8)(Address);
435  }
436  else if (sizeof(AddressType) == 1)
437  {
438      WriteBuffer[0] = (Xuint8)(Address);
439  }
440
441  /*
442  * Put the data in the write buffer following the address
443  */
444  for (Index = 0; Index < ByteCount; Index++)
445  {
446      WriteBuffer[sizeof(Address) + Index] = BufferPtr[Index];
447  }
448
449  /*
450  * Set the address register to the specified address by writing
451  * the address to the device, this must be tried until it succeeds
452  * because a previous write to the device could be pending and it
453  * will not ack until that write is complete
454  */
455  do
456  {

```

Program not running. Click on run icon to start. 0x564 162

X982_11_020707

Figure 11: Running `iic_eeprom_rw_test` in gdb

References

DS434 *OPB IIC Bus Interface (v1.02a)*

XAPP979: *Reference Design : OPB IIC Using the ML403 Evaluation Platform*

ML40x Embedded Development Platform User Guide UG080 (v2.5) May 24, 2006

I2C Bus Specification Version 2.1 January, 2000 Philips Semiconductors

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
3/12/07	1.0	Initial Xilinx release.