IR Thermometer MLX90614

February 16, 2012

Contents

1 Introduction 1

2 I2C Communication 1

2.1 Little Endian 2

2.2 Timing 2

2.3 I2C Address 2

2.4 PEC Byte 3

2.5 PWM Mode 4

3 Reading Temperature 4

4 Revision History 4

1 Introduction

The Melexis MLX90614 non-contact infrared thermometer is a uniquely simple way of measuring temperature at a distance. This document is to help you get I2C communication going using the I2C2PC or BL233B.

2 I2C Communication

The chip uses SMBus, with its attendant timing restrictions.

Important:

- The MLX90614 will *not* work unless I2C messages are completed quickly (ie the time between Start and Stop is less than the timeout).
- The address set in the MLX90614 eeprom is in 7 bit format. So if you set 0x5A to eeprom, you will use 0xB4 with the I2C2PC commands. 0xB4 = shift_left(ox5A).

I will just refer to the bus as I2C.

2.1 Little Endian

Data is returned as 2 byte (16bit) *low byte first* numbers.

2.2 Timing

The SMBus interface is timing dependent. Unless all commands are sent in a tight block, it does not respond. To read it using the BL233B / I2C2PC you should use the ":" command to queue each command. This holds all command characters in the fifo until a linefeed (LF) is received, then executes them all at once. See BL233 datasheet for more details.
To read the IR temperature sensor

```
:S00 07 R 02 P <LF>
```

You must use this ":" mode even if it seems to work. As Windows cannot guarantee when your command characters will be sent, you may have intermittent operation as the delay between characters is uncontrolled.

Only put a single command within the ": .... <LF>" block - it has to fit within the BL233's fifo.

From Realterm, use the *Send* tab. To put the LF on the end either check the "+LF" box, or put "\n" in your string.

Having Problems

Make sure you use RTS/CTS flow control. If the CTS light in Realterm goes out, you ether don't have RTS/CTS, or you have left the <LF> off the end of your ":" commands, or they are too long for the buffer.

2.3 I2C Address

Factory default uses the address 00 (GCA or Global Call Address in I2C parlance). To check what addresses are active use the "Scan Bus" button on the I2C tab.

You can change this in the internal eeprom to any other address. As this is the GCA, you should be careful that other chips in a system do not also respond to this address. Ideally you will change the eeprom setting before putting the part into a system with other I2C or SMBus parts. After you have changed the address, 00 remains active as well.

To change it you:

- Erase the address location, wait 5ms minimum.
- Write the new address. Wait 5ms. The address you send will be 7 bit format. So if you write 5A to the eeprom, the address used by I2C2PC will be B4. Note that the PEC value (E1 below) will be different if the address value than 5A is used.

- Put the part into sleep to activate the new address, (or power it down)
- Wake the part up, and use the new address.

```
:S00 2E 00 00 6F P \n L0005
//Erase Address eeprom
:S00 2E 5A 00 E1 P\n L0005  //Write 5A to eeprom address
:S00 FF F3 P\n  //Send SLEEP Command
O00FE //Put SCL low for sleep state
O00FF O00FD L0010 O00FF
//Wake from Sleep (Bus#1)
:S B4 03 P\n L0100 //Address is now B4. Wake ADC
:S B4 03 R 02 P\n //Read valid data
```

These commands can all be sent as a single string. Spaces are best deleted, they are only for clarity. "\n" is only for use in the Realterm send. It is a linefeed character.

2.3.1 Other Addresses

Here are the command strings for a few other addresses, with the PEC calculated for you.

I2C2PC Address	MLX EEProm value	I2C2PC Command String	PEC
90	48	:S00 2E 48 00 9C P<LF>	9C
92	49	:S00 2E 49 00 89 P<LF>	89
98	4C	:S00 2E 4C 00 C8 P<LF>	C8
9A	4D	:S00 2E 4D 00 DD P<LF>	DD
B0	58	:S00 2E 58 00 CB P<LF>	CB
B2	59	:S00 2E 59 00 DE P<LF>	DE
B4	5A	:S00 2E 5A 00 E1 P<LF>	E1
B6	5B	:S00 2E 5B 00 F4 P<LF>	F4
B8	5C	:S00 2E 58 00 CB P<LF>	9F

2.4 PEC Byte

The PEC is a checksum specified for SMBUS. It is also know as SMBus CRC8.

- You do *not* have to read it. You can just read the two data bytes
- You *do* have to write it.

2.4.1 Using Realterm to Calculate a PEC.

In the example above to write the address eeprom you send this ascii string to the I2C2PC (as text)

```
:S00 2E 5A 00 E1 P
```

The command data here is "2E 5A 00", and the PEC byte is "E1"

- Disconnect the I2C2PC, and set Flow Control to “None”. You are *not* sending a useful string to the I2C2PC, just getting the PEC to display on the terminal screen.
- On “Display” tab, set “Display As” to “Hex+Space”, and set “Half Duplex”
- On “SEND” tab, set “EOL + CRC” and select “SMBUS8” as the CRC type
- Enter just the command data *without* the address byte, PEC byte, as hex numbers. So put the command data into he send tab like this “0x2E 0x5A 0x00”
- Press “Clear” button to clear the terminal window
- Press “Send Numbers”
- The terminal window will show “2E 5A 00 E1” in green. “E1” is the PEC byte.

2.5 PWM Mode

The part can also be configured in eeprom to output a PWM signal on the SDA line. A special pulse on SCL will change it to I2C mode.

PWM Mode is not enabled from the factory. However if you do enable it you will need to precede your commands with a 2 ms low on SCL to switch back to I2C.

If using Bus#1 you pulse SCL low.

```
O00FE L0002 O00FF
```

3 Reading Temperature

Substitute the address of your sensor below for the GCA (00)

Ambient (package) Temperature is 06

```
:S00 07 R 02 P <LF>
```

IR Sensor is 07

```
:S00 07 R 02 P <LF>
```

To read both. Note that spaces are ignored by the BL233B.

```
:S00 06 R02P, W06R02P <LF>
```

4 Revision History

Rev	Date	Changes
0	14 Feb 2012	First Release
1	16 Feb 2010	Clarified addressing, Added samples for other addresses
2		
3		