CC32xx I2C Application

Overview

The Inter-Integrated Circuit (I2C) bus provides bi-directional data transfer through a two-wire design (a serial data line SDA and a serial clock line SCL),and interfaces to external I2C devices.On the LaunchPad there are two sensors connected over the I2C namely Accelerometer and Temperature sensor. The I2C lines are also brought

out on the header for further expansion (to connect to sensors on booster packs).

Application details

The objective of this application is act as an I2C diagnostic tool. The demo application is a generic implementation that allows the user to communicate with any I2C device over the lines.

The features supported by the application are:

- **I2C write:** Performs write operation to the specifed I2C device address. User also needs to specify if the stop bit needs to be set or not (To support specifying and reading from a particular register).
- **I2C Read:** Performs I2C read from the specified I2C device address.
- **I2C Readreg:** Is a combination of I2C write (without stop bit) + I2C read. This is useful to get register values from the specified register offset from a I2C device.

This command assumes the register offset to be 1 byte.

- **I2C Writereg:** Performs an I2C writes from the specified offset register. This command assumes the register offset to be 1 byte.

💡 **Note:** The write and read are core APIs that follow the I2C specification for its operation. They are generic and can be used to communicate with any I2C device.

💡 **Note:** The readreg and writereg are ready to use APIsthat internally invoke the read and write functionality. These have been provided for ease of use and is not guaranteed to work for all I2C devices due to some assumptions made.

Source Files briefly explained

- **main**: - Prompts and services the user options by invoking the i2c_if APIs.
- **i2c_if**: - I2C access interface APIs. This works in a polled mode of operation.
- **pinmux**: - Pinmux configurations as required by the application.
- **startup_***: - Tool specific vector table implementation.
- **uart_if**: - To display status information over the UART

Usage

💡 **Note:** For this application to work, one needs to populate jumpers at j2 and j3 on LP 3.x

1. Setup a serial communication application (HyperTerminal/TeraTerm). For detail info visit Terminal setup

    On the host PC, open a hyperterminal, with the following settings

    - **Port:** 2nd enumerated COM port
    - **Baud rate:** 115200
    - **Data:** 8 bit
    - **Parity:** None
    - **Stop:** 1 bit
    - **Flow control:** None

2. Run the reference application (Flashing the bin/IAR/CCS).

    - Open the Project as mentioned in the 'docs\CC3200-Getting Started Guide.pdf'
    - Build and download the application to the board

3. On the Hyperterminal, a prompt appears

    - The I2C commands need to be issued and the results can be seen

CC3200 I2C Example Usage

Commands:

> write <dev_addr> <wrlen> <<byte0> [<byte1> ...]> <stop>
>
> read <dev_addr> <rdlen>
>
> writereg <dev_addr> <reg_offset> <wrlen> <<byte0> [<byte1> ...]>
>
> readreg <dev_addr> <reg_offset> <rdlen>

Parameters:

dev_addr - slave address of the i2c device, a hex value preceeded by '0x'

   reg_offset - register address in the i2c device, a hex value preceeded by '0x'

wrlen - number of bytes to be written, a decimal value

   rdlen - number of bytes to be read, a decimal value

bytex - value of the data to be written, a hex value preceeded by '0x'

stop - number of stop bits, 0 or 1

💡 **Note:** The user can determine the device address of a particular component by looking at its respective datasheet.

- The temperature sensor used is **tmp006**, and its device address (dev_addr) is **0x41**
- The accelerometer sensor used is **bma222**, and its device address (dev_addr) is **0x18**

For instance

- To read the manufacturer id register (0xFE) of the temperature sensor (0x41), the command will be
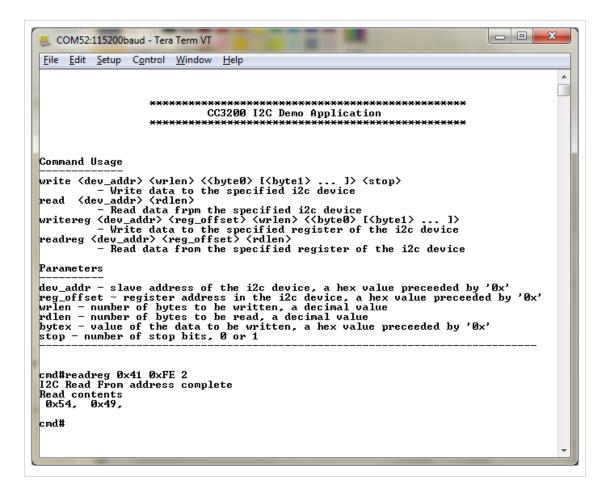
```
readreg 0x41 0xFE 2
```

- To read the "who am i" register (0x00) of the accelerometer sensor (0x18), the command will be

```
readreg 0x18 0x00 1
```

Snapshot of the Hyperterminal when the application starts.

```
CC3200 I2C Demo Application
**********************************************

Command Usage
-------------
write <dev_addr> <wrlen> <<byte0> [<byte1> ... ]> <stop>
          - Write data to the specified i2c device
read  <dev_addr> <rdlen>
          - Read data frpm the specified i2c device
writereg <dev_addr> <reg_offset> <wrlen> <<byte0> [<byte1> ... ]>
          - Write data to the specified register of the i2c device
readreg <dev_addr> <reg_offset> <rdlen>
          - Read data from the specified register of the i2c device

Parameters
----------
dev_addr - slave address of the i2c device, a hex value preceeded by '0x'
reg_offset - register address in the i2c device, a hex value preceeded by '0x'
wrlen - number of bytes to be written, a decimal value
rdlen - number of bytes to be read, a decimal value
bytex - value of the data to be written, a hex value preceeded by '0x'
stop - number of stop bits, 0 or 1
----------------------------------------------------------------------


cmd#readreg 0x41 0xFE 2
I2C Read From address complete
Read contents
 0x54,  0x49,

cmd#
```

Limitations/Known Issues

1. This example exercise only standard speeds(100Kbps and 400Kbps) for I2C clock. High speed(1 Mbps and 3.3 Mbps) I2C clocks are supported in driverlib, but not exercised in this example.

2. Populating the jumpers J2 and J3 will turn the LED D5 and D6 on and hence will not be available for use.

Article Sources and Contributors

CC32xx I2C Application *Source*: http://processors.wiki.ti.com/index.php?oldid=178459 *Contributors*: A0221015, Codycooke, Jitgupta, Malokyle

Image Sources, Licenses and Contributors

File:Cc31xx cc32xx return home.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Cc31xx_cc32xx_return_home.png *License*: unknown *Contributors*: A0221015

File:Cc32xx return sample apps.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Cc32xx_return_sample_apps.png *License*: unknown *Contributors*: A0221015

File:Light_bulb_icon.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Light_bulb_icon.png *License*: unknown *Contributors*: DanRinkes, PagePusher

Image:I2c demo.png *Source*: http://processors.wiki.ti.com/index.php?title=File:I2c_demo.png *License*: unknown *Contributors*: Codycooke

License

a. This License and the rights granted hereunder will terminate automatically upon any breach by You of the terms of this License. Individuals or entities who have received Adaptations or Collections from You under this License, however, will not have their licenses terminated provided such individuals or entities remain in full compliance with those licenses. Sections 1, 2, 5, 6, 7, and 8 will survive any termination of this License.

b. Subject to the above terms and conditions, the license granted here is perpetual (for the duration of the applicable copyright in the Work). Notwithstanding the above, Licensor reserves the right to release the Work under different license terms or to stop distributing the Work at any time; provided, however that any such election will not serve to withdraw this License (or any other license that has been, or is required to be, granted under the terms of this License), and this License will continue in full force and effect unless terminated as stated above.

8. Miscellaneous

a. Each time You Distribute or Publicly Perform the Work or a Collection, the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.

b. Each time You Distribute or Publicly Perform an Adaptation, Licensor offers to the recipient a license to the original Work on the same terms and conditions as the license granted to You under this License.

c. If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.

d. No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.

e. This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that may appear in any communication from You. This License may not be modified without the mutual written agreement of the Licensor and You.

f. The rights granted under, and the subject matter referenced, in this License were drafted utilizing the terminology of the Berne Convention for the Protection of Literary and Artistic Works (as amended on September 28, 1979), the Rome Convention of 1961, the WIPO Copyright Treaty of 1996, the WIPO Performances and Phonograms Treaty of 1996 and the Universal Copyright Convention (as revised on July 24, 1971). These rights and subject matter take effect in the relevant jurisdiction in which the License terms are sought to be enforced according to the corresponding provisions of the implementation of those treaty provisions in the applicable national law. If the standard suite of rights granted under applicable copyright law includes additional rights not granted under this License, such additional rights are deemed to be included in the License; this License is not intended to restrict the license of any rights under applicable law.