通过一条command就可以在i2c bus上create one i2c device或delete one i2c device(created from user-space)

in Documentation/i2c/instantiating-devices.txt

Method 4: Instantiate from user-space

In general, the kernel should know which I2C devices are connected and what addresses they live at. However, in certain cases, it does not, so a sysfs interface was added to let the user provide the information. This interface is made of 2 attribute files which are created in every I2C bus directory: new_device and delete_device. Both files are write only and you must write the right parameters to them in order to properly instantiate, respectively delete, an I2C device.

File new_device takes 2 parameters: the name of the I2C device (a string) and the address of the I2C device (a number, typically expressed in hexadecimal starting with 0x, but can also be expressed in decimal.)

File delete_device takes a single parameter: the address of the I2C device. As no two devices can live at the same address on a given I2C segment, the address is sufficient to uniquely identify the device to be deleted.

Example:

echo eeprom 0x50 > /sys/bus/i2c/devices/i2c-3/new_device

While this interface should only be used when in-kernel device declaration
can't be done, there is a variety of cases where it can be helpful:

* The I2C driver usually detects devices (method 3 above) but the bus
  segment your device lives on doesn't have the proper class bit set and
  thus detection doesn't trigger.
* The I2C driver usually detects devices, but your device lives at an
  unexpected address.
* The I2C driver usually detects devices, but your device is not detected,
  either because the detection routine is too strict, or because your
  device is not officially supported yet but you know it is compatible.
* You are developing a driver on a test board, where you soldered the I2C
  device yourself.

This interface is a replacement for the force_* module parameters some I2C
drivers implement. Being implemented in i2c-core rather than in each
device driver individually, it is much more efficient, and also has the
advantage that you do not have to reload the driver to change a setting.
You can also instantiate the device before the driver is loaded or even
available, and you don't need to know what driver the device needs.

在G2 LSP中 Touch screen device就是挂在I2C4上的device

```
        pxai2c4: i2c@d4033000 {
            pinctrl-0 = <&i2c1_pins>;
            pinctrl-names = "default";
            status = "okay";
            polytouch: edt-ft5x06@38 {
                compatible = "edt,edt-ft5x06";
                reg = <0x38>;
                pinctrl-names = "default";
                interrupt-parent = <&gpio0>;
                interrupts = <35 0>;
                num-x = <1024>;
                num-y = <600>;
                invert-y = <1>;
                invert-x = <0>;
                reset-gpios = <&gpio0 36 0>;
            };
        };
```

目前该device is not work。可以注释掉该i2c device，然后用这里的方法去create 0x38的device，然后使用

i2c-tools package中的i2cdetect来测试该device是否有回应。

create i2c device

```
root@granite2:~# echo edt-ft5x06 0x38 > /sys/bus/i2c/devices/i2c-3/new_device
i2c i2c-3: new_device: Instantiated device edt-ft5x06 at 0x38
```

edt-ft5x06是i2c device的name

0x38是address

delete i2c device

```
root@granite2:~# echo 0x38 > /sys/bus/i2c/devices/i2c-3/delete_device
i2c i2c-3: delete_device: Deleting device edt-ft5x06 at 0x38
```

---

test i2c device

root@granite2:~# ./i2cdetect -l

i2c-0 i2c pxa_i2c-i2c I2C adapter

i2c-1 i2c pxa_i2c-i2c I2C adapter

i2c-2 i2c pxa_i2c-i2c I2C adapter

i2c-3 i2c pxa_i2c-i2c I2C adapter

i2c-4 i2c pxa_i2c-i2c I2C adapter

i2c-5 i2c pxa_i2c-i2c I2C adapter

root@granite2:~# modprobe i2c-dev

root@granite2:~# ./i2cdetect -y 3 0x30 0x40

0 1 2 3 4 5 6 7 8 9 a b c d e f

00:

10:

20:

30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

40: --

50:

60:

70:

没找到0x38的device ???

root@granite2:~# ./i2cdetect -y 0 0x10 0x60

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:
10: -- -- -- -- -- -- -- -- 18 19 -- 1b -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- 33 -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: 50 -- -- 53 -- -- -- -- -- -- -- -- -- -- -- --
60: --
70:
```

在i2c-0上却搜到了这么多设备

Gemstone2 board上i2c4 module不工作的原因如下：

季宾在帮我焊接probe point时焊接有问题。

换了一块board（原来i2c4上clock pin连接错的地方rework后），i2c4 work

root@granite2:~# ./i2cdetect -y 3 0x30 0x40

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:
```

10:

20:

30: -- -- -- -- -- -- -- -- 38 -- -- -- -- -- -- --

40: --

50:

60:

70:

i2cdetect能很顺利的detect 0x38 address.