GM8126

I2C

User Guide
Rev.: 1.0
Issue Date: December 2010

REVISION HISTORY

GM8126 I2C User Guide

Date	Rev.	From	To
Dec. 2010	1.0	-	Original

Printed in Taiwan 2010

Grain Media, Inc.
5F, No. 5, Li-Hsin Road III, Hsinchu Science Park, Hsinchu City, Taiwan 300, R.O.C.

Grain Media's home page can be found at:
http://www.grain-media.com

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

I2C Overview

I2C (Inter-Integrated Circuit) might be the most popular interface in an embedded system. It uses only three lines to connect the external devices and main chip for exchanging the control messages and data.

GM8126 has one built-in I2C IP. It can only be an I2C master. If the external device needs to asynchronously send some messages to the main chip, users may use the polling method or an extra pin to indicate the situation of the main chip. Obviously, the latter is more efficient, but one GPIO pin will be wasted.

GM8126 I2C provides rich features and is capable of handling various types of I2C protocols. In the following sections, the configuration of the Linux kernel will be introduced so that the functions will work properly.

Chapter 2

Configuration Items in Linux Kernel

Before using the provided I2C functions, users should make sure that the following kernel options are enabled.

Device Drivers

- I2C support
  - I2C device interface
  - Auto-select pertinent helper modules
  - I2C hardware bus support
    - GM FTIIC010 I2C controller
    - GM I2C uses the interrupt mode.

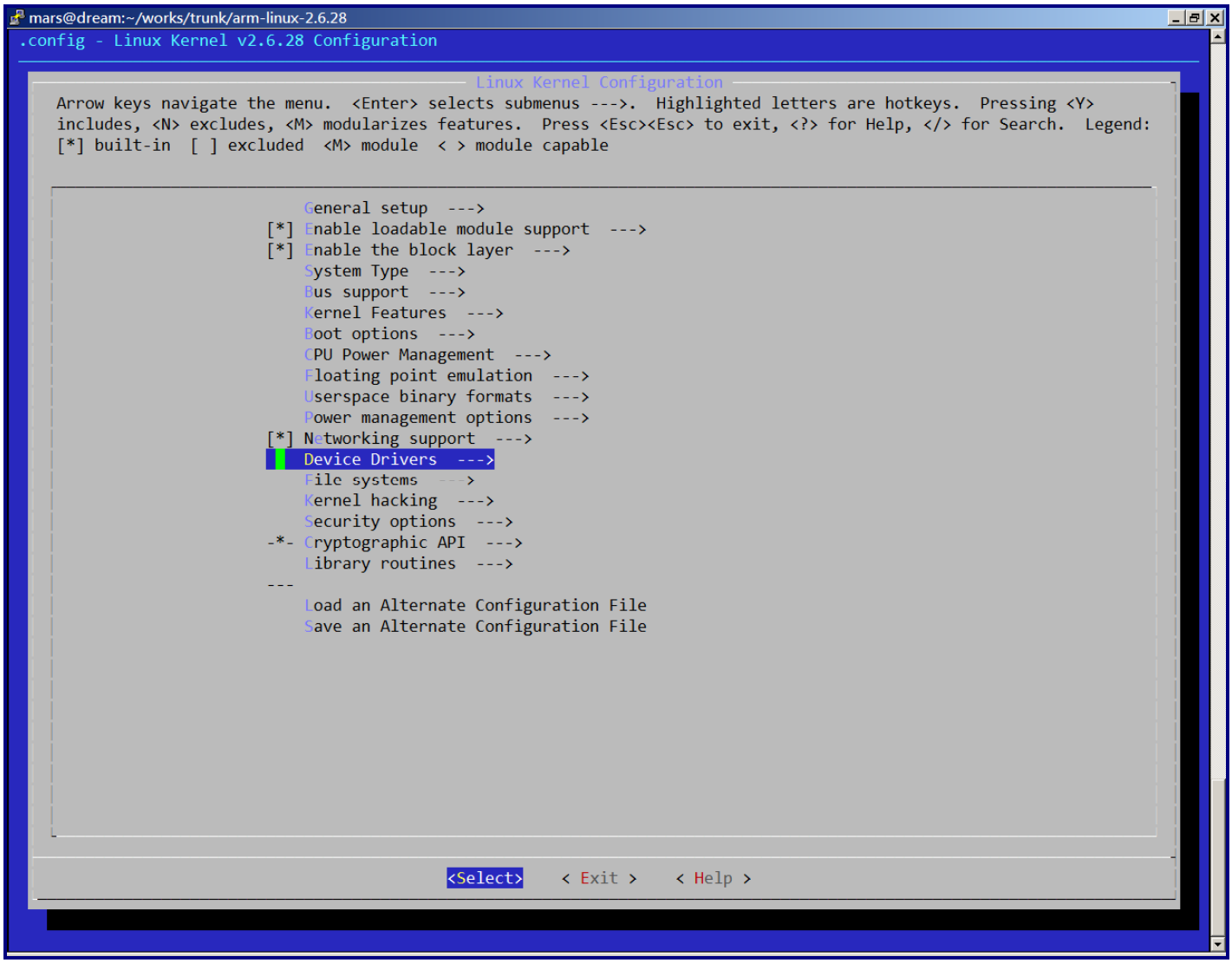

Figure 2-1. Device Drivers Entry

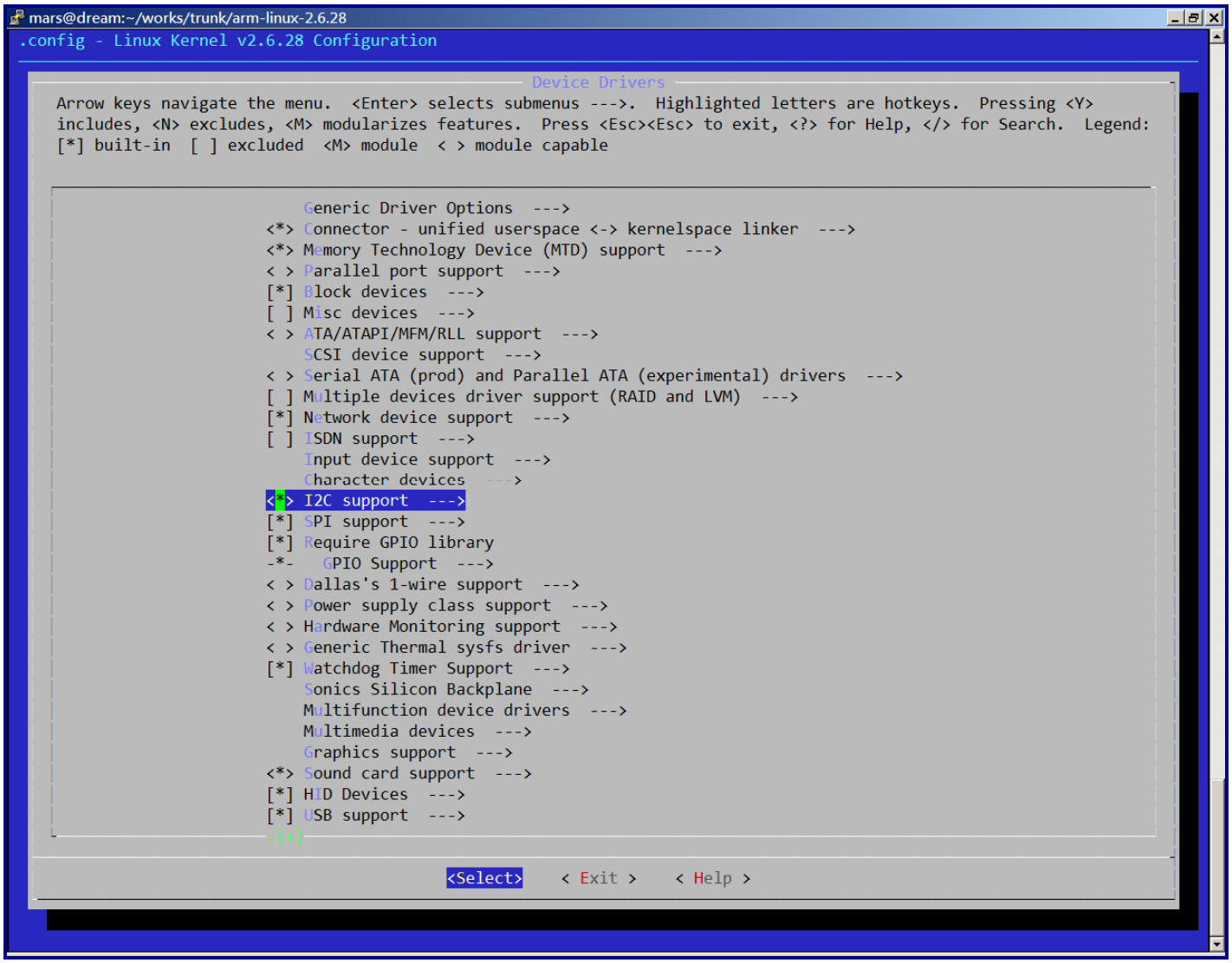

Figure 2-2. Enable I2C Category

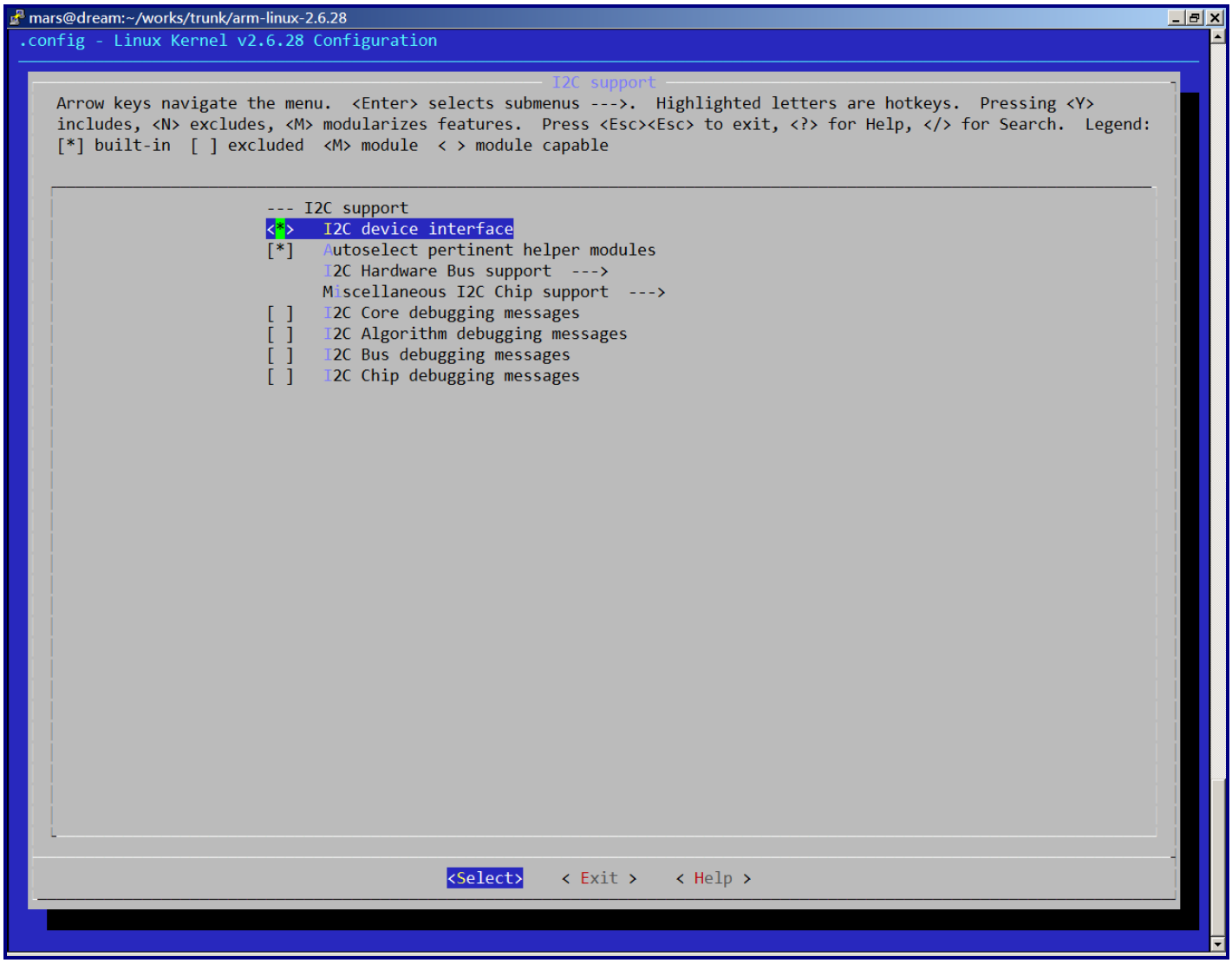

Figure 2-3. Enable Necessary I2C Features Kernel Provided

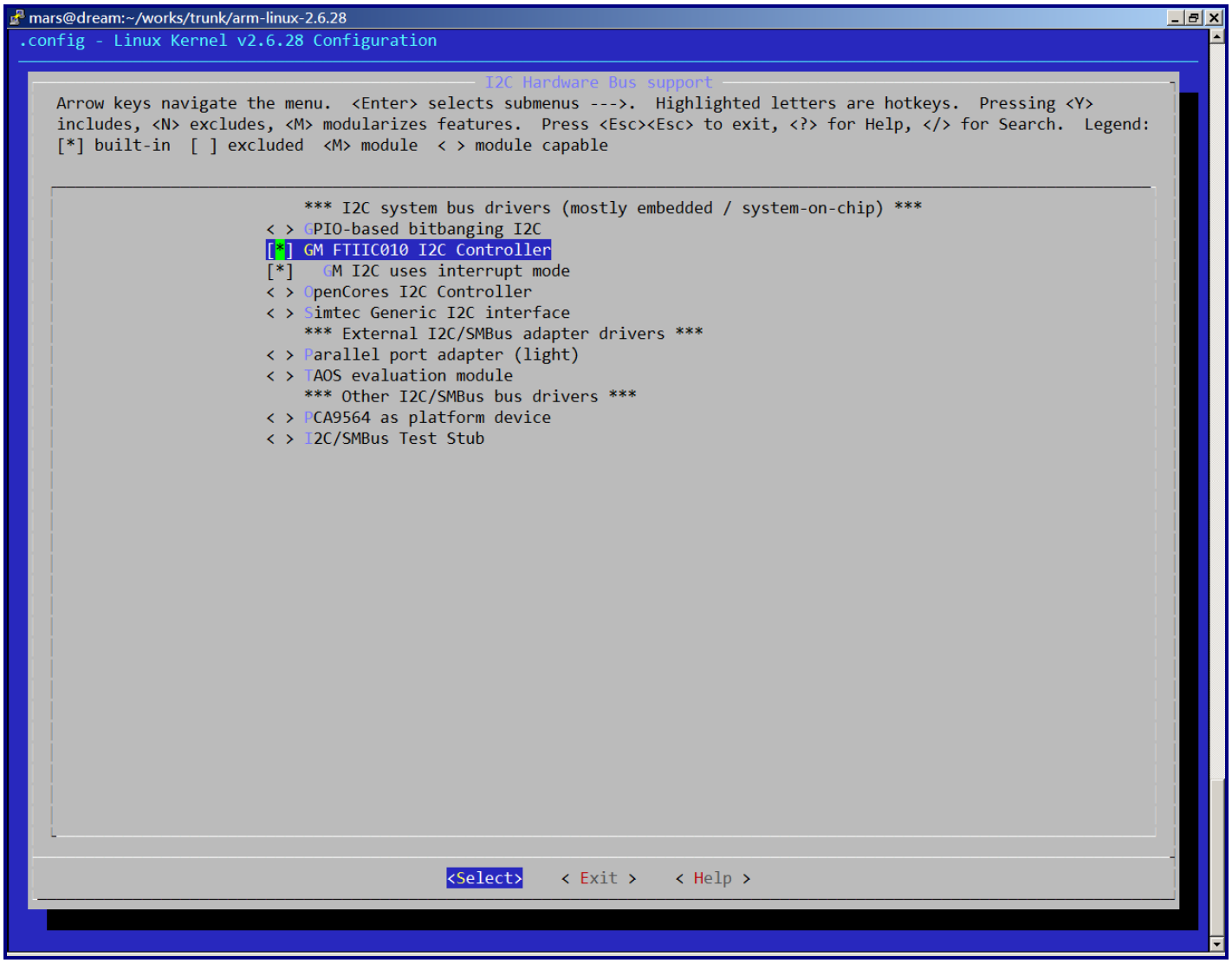

Figure 2-4. Enable Specific Driver of GM8126

Users should pay special attention to the item, “GM I2C uses interrupt mode”. The GM8126 I2C IP can use this item to trigger an interrupt when receiving an ACK from devices instead of waiting to check the status of the related register; thus, users can save a lot CPU time. These useful configuration items can ease the efforts of enabling various functions.

The above options are included in the default configuration file. Users may check them in a case of unexpected condition.

```
/ # ls /sys/class/i2c-*
/sys/class/i2c-adapter:
i2c-0

/sys/class/i2c-dev:
i2c-0
/ #
```

Figure 2-5. I2C Information in /sys

Chapter 3

Utilities in userspace

The userspce I2C utilities are addressed in this section. Users can refer to Figure 3-1 to know how kernel layers the I2C related components.

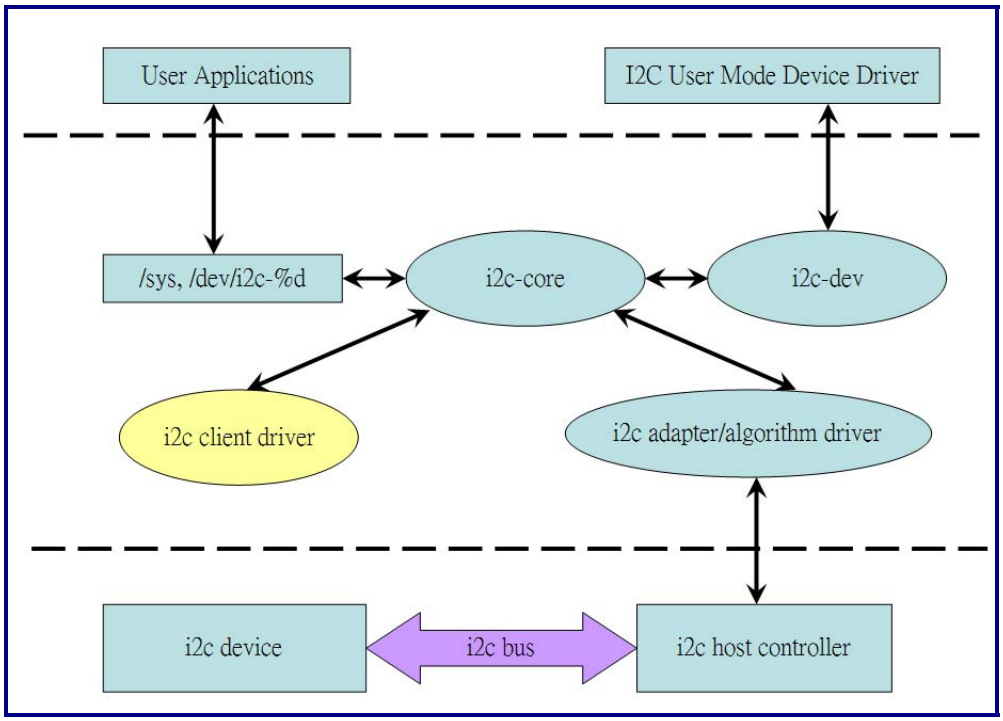

Figure 3-1. I2C Architecture in Linux

After configuring the related I2C options in the previous section, users have turned on the I2C core, I2C device, and I2C adapter/algorithm driver. The I2C core and I2C device drivers are the middleware in the Linux subsystem. The I2C core is the main component to provide the interfaces to other parts and the I2C device provides a series of APIs for developing the userspace I2C driver. The I2C adapter/algorithm driver controls the host driver, which is implemented in i2c-ftiic010.c. The I2C client driver uses the I2C core interfaces to send and receive the I2C messages. In most cases, users should only pay attention to the I2C client part.

i2c_access is included in the Grand Media SDK. Users can check it from the usr/i2c_program/i2c_access directory. It uses the standard interfaces Linux provided to send and receive the I2C messages in the userspace utility. Figure 3-2 shows the usage of i2c_access.

```
/ # i2c_access –help

There are four usages:
1. i2c_access I2C_ADDR w [B][B]...[B][B] r [B]\n");
2. i2c_access nostop I2C_ADDR w [B][B]...[B][B] r [B]\n");
3. i2c_access I2C_ADDR w [B][B]...[B][B]\n");
4. i2c_access I2C_ADDR r [B]\n");

I2c_ADDR: the I2C address of the specific device you want to talk, 7bits format\n");
nostop:   the first method will insert a stop between write cmd and read cmd, nostop won't
r/w:       specify r, w to read or write the following data byte
[B]:       when writing, this means 8 bits data byte
           when reading, this means the number of byte will be read
```

Figure 3-2. i2c_access Usage

Please note that Grain Media provides a version of the write-read command without the stop condition. The following are more real examples:

//Write three bytes to a device with the I2C address 0x68 in 7-bit height

 # i2c_access 0x68 w 0x00 0x01 0x02

//Read three bytes from a register with offset 0x01

 # i2c_access 0x68 w 0x01 r 0x03

//It is the same as the previous one without stopping at the write-read command.

 # i2c_access nostop 0x68 w 0x01 r 0x03

Chapter 4

Related Files

The following directory contains all the details that the Linux kernel provides. Users will know how Linux constructs the I2C sybsystem.

- arm-linux-2.6.28/linux-2.6.28-fa/Documentation/i2c/

The following two files set the clock and device definitions of in GM8126.

- arm-linux-2.6.28/linux-2.6.28-fa/arch/arm/mach-GM/platform-GM8126/pmu.c
- arm-linux-2.6.28/linux-2.6.28-fa/arch/arm/mach-GM/platform-GM8126/platform.c

This implements the Linux I2C subsystem.

- arm-linux-2.6.28/linux-2.6.28-fa/drivers/i2c/

This is the specific control driver from Grain Media.

- arm-linux-2.6.28/linux-2.6.28-fa/drivers/i2c/busses/i2c-ftiic010.c

This implements i2c_access

- arm-linux-2.6.28/user/i2c_program/i2c_access