M.2 Adapter and Power Control Card

Added by STEPHEN BOWERS, last edited by DONGYUE JIAO on Dec 08, 2014

General explanation:

This describes how to load and use the adapter card for use with the M.2's.

For Adapter card specs and more description, such as "Hardware User's Guide", go here:

http://collab.micron.com/corp/ARD/hardware/Wiki%20Pages%20ESSD/PCIe%20M.2%20SSD%20Adapter.aspx

Location of Adapter firmware code and python scripts:

stash -> SSD Client Test Automation -> jenkins-scripts -> adapter_card

(1) firmware (2) os_code

Tools and software required:

1. WinAVR (Freq of board is different than Arduino Programmer, so we are using this instead)

webpage:

Please download and install WinAVR: http://winavr.sourceforge.net/

compile:

Find the "sample" folder under WinAVR installation path, duplicate and modify the "Makefile" for your own project:

(1) modify the MCU by setting "MCU = atmega328p"

(2) modify the crystal frequency by setting "F_CPU = 8000000"

(3) modify the target file by setting "TARGET = hex_file_name_after_compile"

(4) modify the output file format after compile by setting "FORMAT = ihex"

upload the program into the micro:

Open a windows prompt, reach the WinAVR installation path and find avrdude.exe. Then run the following command:

avrdude.exe -c usbtiny -p m328p -U flash:w:path\to\hex\C\file\file_name.hex:i

(1) usbtiny is the programmer we use;

(2) m328p is the chip;

(3) the -U argument is the path for hex file;

2. FTDI 4232 USB to UART chip (Install Driver):

For Windows:

The VCP driver is required to run FTDI 4232 as a general COM port: http://www.ftdichip.com/Drivers/VCP.htm

After installing the VCP driver, four COM ports will show up in "Device Manager", please choose the 2nd one(COM6 among COM5~8) of these 4 COM ports as the USART communication channel. The setting of the specific COM port for communication is in Python program.'

For Linux:

Since the FTDI VCP driver is built into the Linux kernel, no VCP driver installation is required. When connecting the chip, the VCP drivers will appear as /dev/ttyUSBx.

For Cygwin:

In order to run this script in cygwin prompt, do not need to disable the corresponding serial port in windows. Because the disabled serial port can not show up in cygwin as well.

3. Other software/package (using Python for script development):

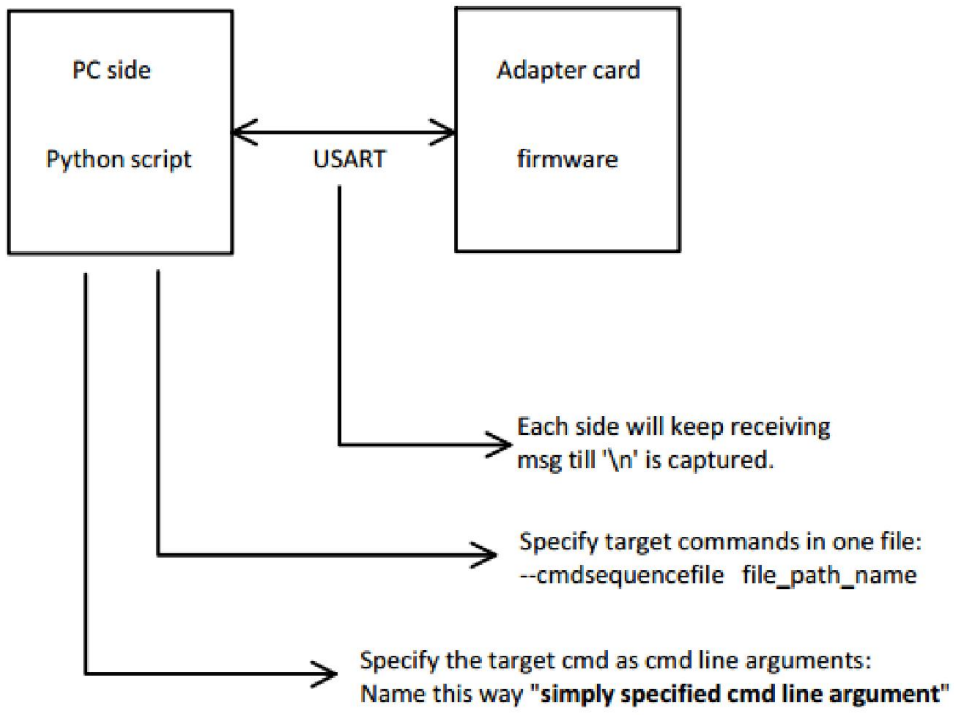
python2.7 command prompt **serial package** is required, and it is located in our git repo as well.

How does the python program work with the firmware program together:

Brief idea

PC(Python program) will communicate with the micro via the USART port for sending command out and receiving data back. The micro will run the command receiving from the PC side, and send data/string back. The feedback in for format of MSbit to LSBit order. Both Python program and firmware program keep waiting until they receive a '\n' character.

The communication diagram and corresponding note following as:



Specify the serial port by setting "--comport"

For windows, in format of: COMx

For linux, in format of: /dev/ttyUSB1

For cygwin, in format of: /dev/ttySx

Run script in "simply specified cmd line argument" mode

(1) available command line arguments:

command parameter	details	note
--poweron	please refer the other table for --cmdsequencefile arg	If multiple command arguments are specified, they will execute in predefined order. The execution order sequence is from top to bottom of left most column, which means the higher the command in the left most column is, the earlier the command got execution. For example, if the cmd line arg specifies as "--poweron --shuntvolt --turnOFF --current --readconfig", the actual execution sequence is "--poweron --readconfig --shuntvolt --current --turnOFF". If you want to change the execution sequence, manually modification is required in python script.
--setconfig		
--setcalib		
--readconfig		
--readcalib		
--shuntvolt		
--busvolt		
--power		
--current		
--turnOFF		

(2) example of "simply specified cmd line argument":

python2.7 adapter_card_power_onoff.py --comport COM6 --poweron --current --busvolt --current --setconfig --setcalib --turnOFF

In the above command, serial port "COM6" is in use.

And the commands will be executed in the following order: poweron, setconfig, setcalib, busvolt, current, turnOFF.

Although "–current" has been specified twice, but only one will be executed.

Run script in command file mode by issuing commands in a separate file

(1) available command that can be included in the separate file:

All available commands are in format of "0xEExxxx0xFF".

Category	Command	Description	Note
For power regulator to control	**0xEEpoweron0xFF**	**Turn on Power to M.2 Drive**	**The LED can tell the status of the power regulator, on or off.**

SSD drive power supply	**0xEEturnOFF0xFF**	**Turn off the Power to M.2 Drive**	
Register set/read commands	**0xEEsetconfig0xFF**	**Set configuration register of INA209**	**The value set to config reg has been defined in the .c file. Because seldom this register required update, so manually modification is the way to do the change in .c file.**
	0xEEreadconfig0xFF	**Read configuration register value out**	**For verification.**
	0xEEsetcalib0xFF	**Set calibration register of INA209**	**The value set to calib reg has also been defined in the .c file. Manually modification is required in .c file in order to change the value of this register.**
	0xEEreadcalib0xFF	**Read Calibration register value out**	**For verification.**
Data register access command	**0xEEshuntvolt0xFF**	**Read shunt volt out**	**MSBit comes out first, then convert to decimal, multiply the precision, and display in terminal.**
	0xEEbusvolt0xFF	**Read bus volt out**	**MSBit comes out first, then convert to decimal, multiply the precision, and display in terminal.**
	0xEEpower0xFF	**Read power consumption out**	**MSBit comes out first, then convert to decimal, multiply the precision, and display in terminal.**
	0xEEcurrent0xFF	**Read current out**	**MSBit comes out first, then convert to decimal, multiply the precision, and display in terminal.**

(2) example to run the script with a specified command file:

python2.7 adapter_card_power_onoff.py --cmdsequencefile ../cmd_seq.txt

The content of cmd_seq.txt following as:

0xEEpoweron0xFF
#need to set config reg and calib reg
0xEEsetconfig0xFF
0xEEsetcalib0xFF

#this is an empty line to show comment

0xEEreadconfig0xFF
0xEEreadcalib0xFF

.....

Other helpful links:

SPI upload programmer introduction: https://learn.sparkfun.com/tutorials/pocket-avr-programmer-hookup-guide

INA209: http://www.ti.com/lit/ds/symlink/ina209.pdf

ATmega328P: http://www.atmel.com/images/doc8161.pdf

FTDI4232: http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT4232H.pdf

I2C introduction: http://www.robot-electronics.co.uk/acatalog/I2C_Tutorial.html

Like Be the first to like this