eMMC測試程式架構

由於在linux寫UI程式有點麻煩加上大家使用習慣仍是windows所以我的想法是使用網頁的方式來做UI的介面，只要透過Browser連上網路就可以控制我們的測試程式。



軟體架構如下所示



在PC端我們透過Browser傳request給平台上的server，當平台收到request透過PHP呼叫 C程式碼編譯的測試程式，當測試完成後再將結果回傳給 server 然後等待client來接收測試結果。

工作項目:

1. 使用debugfs與mmc\_test溝通。
   1. 在mms\_test 上的debugfs加入我們的變數
   2. 傳遞資料到MMC\_TEST
2. 整理driver提供Interface。
   1. Erase
   2. eMMC command
   3. Switch partition
   4. single read/write
   5. multi read/write
3. 需在平台上架設一個Apache HTTP伺服器。
4. 利用PHP透過C++/JAVA與平台溝通。
5. 使用JAVA透過JNI與 C 編譯程式溝通。

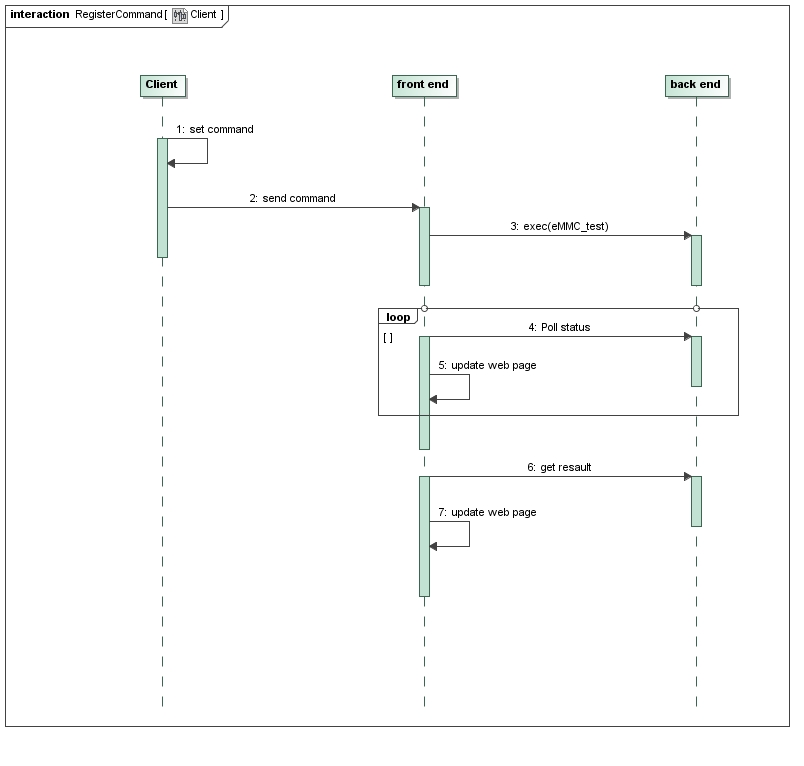
本系統所提供的服務

1. 可執行一個command sequence(一串eMMC 命令序列),並顯示結果
2. 執行read command
3. 執行 write command
4. 執行 erase command
5. 執行setting command
   * Clock
   * signal\_voltage
   * bus\_width
   * bus\_mode

**實作部分:**

**From end**

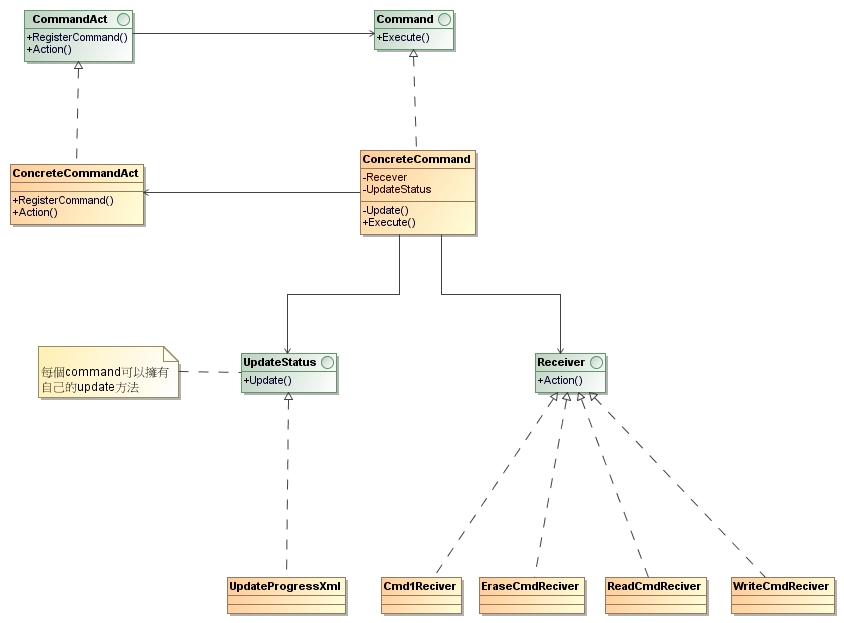
循序圖



**Back end**

程式架構類別圖:

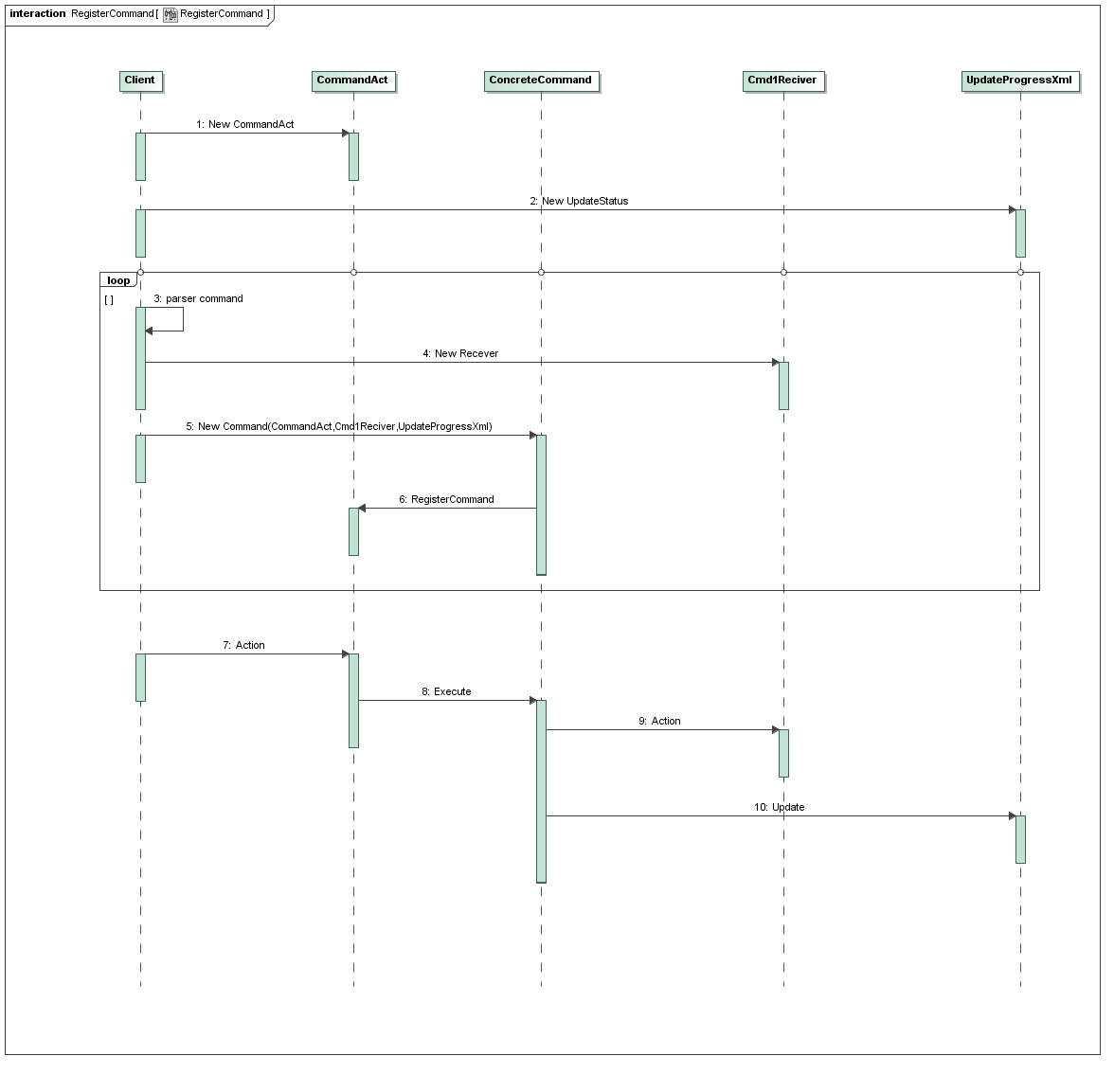
考慮到需要執行程式序列以及command的擴充，在此採用採用觀察者模式,以及命令模式混合模式。



循序圖:

程式在執行一開始會parser封裝好的command xml檔，並將parser的結果初始化為ConcreteCommand物件。而ConcreteCommand會自己註冊到CommandAct物件中，當初始化完畢，再執行CommandAct.Action，執行使用者的request。

每個command 執行完後會產生一個result xml，from end polling到後再將結果秀在網頁上。



**Driver layer**

使用debugfs 作為與mmc\_test溝通的橋樑,將我們的command傳入kernel space做法如下:

待補........................

其中mmc\_test

所提供的API如下

**Setting:**

1. mmc\_set\_blockcount
2. mmc\_set\_blocklen
3. mmc\_set\_clock
4. mmc\_set\_signal\_voltage
5. mmc\_set\_bus\_width
6. mmc\_set\_bus\_mode

**Send command:**

1. mmc\_wait\_for\_req :參數為mrq
2. mmc\_wait\_for\_cmd:參數為cmd

**Reset:**

1. mmc\_do\_hw\_reset:實測試無法work，嘗試將他打通

**Read/Write command:**

1. mmc\_test\_area\_transfer :single/multi block read/write

**erase**

1. mmc\_test\_area\_erase : 對一個區間erase

**Trim**

1. mmc\_test\_profile\_seq\_trim\_perf :對一區間trim以及測其效能

**LAMP setup**

Step:

1. Install LAMP (linux + apache + mysql + php) in the eMMC Device

(1) vli:~$ sudo apt-get install lamp-server^

(^ represents immediatelly execution)

(2) Enter the root password for mysql twice.

(3) vli:~$ sudo apt-get install libapache2-mod-auth-mysql

(4) We can find the index.html file in /var/www/html

(5) We can create the folder (ex: emmc\_0.1)

in /var/www/html and put the html or php files in it.

-------------------------------------------------------------------------------------------

2. Write http code (send.html)

(1) Description: Send two variables (a, b) which can be manipulated by php (receive.php).

(2) Code:

<html>

<head>

<title>Transmission of two values test</title>

</head>

<body>

<h1>XDD</h1>

<form method="post" action="receive.php">

<input type="text" name="a"><br/>

<input type="text" name="b">

<input type="submit" value="Send">

</form>

</body>

</html>

-------------------------------------------------------------------------------------------

3. Write php code (receive.php)

(1) Description: Receive the passing variables (a, b) which is generated by html (send.html)

(2) Code:

<html>

<head>

<title>Receive</title>

</head>

<body>

<?php

$a = $\_POST["a"];

$b = $\_POST["b"];

echo "<h1>Value a: " . $a ."</h1>";

echo "<h1>Value b: " . $b ."</h1>";

echo exec ("/var/www/html/emmc\_0.1/add $a $b");

?>

</body>

</html>

-------------------------------------------------------------------------------------------

(gcc)

4.1. Write c code (main.c ---------> add)

(1) Description: Compute the sum of the two variable and return a value.

(2) Code: main.c

#include <stdio.h>

int add(char \*a, char \*b) {

return atoi(a)+atoi(b);

}

int main (int argc, char \* argv[]) {

if (argc==0) {

return -1;

} else {

printf ("The result is :%d\n", add (argv[1], argv[2]));

}

return 0;

}

(3) vli:~$ gcc -o add main.c

(4) In the receive.php:

echo exec ("/var/www/html/emmc\_0.1/add $a $b");

-------------------------------------------------------------------------------------------

Optional:

4.2. Write java code

(1) apt-get install openjdk-7-jdk

(2) Code: hello.java

public class hello {

public static void main (String[] args) {

int value = Integer.valueOf(args[0]) + Integer.valueOf(args[1]);

System.out.println (value);

}

}

(3) vli:~$ javac hello.java

(4) In the receive.php:

echo exec ("java hello $a $b");