MK14keys webpages

Some setup instructions for using the webserver on a raspberry pi.

Things that need to be done

1. Install the raspberry pi operating system.
2. Install the lighttpd web server
3. Install and enable the PHP modules
4. Configure the "php.ini" File
5. Configure the mimetype on the webserver to allow .hex text files
6. Copy the MK14keys files to the webserver directory
7. Create the mk14 directory
8. Allow the webserver to access the GPIO pins
9. Test the process.

1. Install the raspberry pi operating system.

I started the process by installing a copy of the Raspbian version 11 (bullseye) on a new SD card, the order I did things are not necessarily the best but worked for me.
If you already have a working pi then it could well be that it will work fine.

I used the “Raspberry Pi Imager” and installed the “Raspberry Pi OS Lite (32=bit)” operating system as I normally connect to the pi using SSH.

Active the ssh interface by placing a file called ssh on the boot partition before transferring it to the pi.
Reference https://pimylifeup.com/raspberry-pi-enable-ssh-boot/

Now boot the raspberry pi (it will need a wired network connection not wifi) and you should have a system called raspberrypi that you can ping.
Connect using ssh – on linux that is ssh pi@raspberrypi , or use an ssh console, i.e. putty, on a windows system.

Once logged into raspberry pi use sudo raspi-config
   change it’s name to mk14pi
   change the password for the user pi
   expand the file system to fit the sd card.
Don’t forget to update the operating system before continuing.

```
sudo apt-get update
sudo apt-get upgrade
```

Now reboot the raspberry pi.

2. Install the lighttpd web server

We need a web server for this to work. I used the lighttpd web server since I had used it before when I had installed pi-hole.
If you have a web server already running then skip to the next step.

This is an update command

```
        sudo apt install lighttpd
```

which should run and install the webserver.

Use a web browser to go to http://mk14pi should show the holding page.

The folder locations are

```
    /var/www/html – is the default root folder for Lighttpd.
    /etc/lighttpd/ – is the default folder for Lighttpd configuration files.
```

See https://www.tecmint.com/install-lighttpd-in-ubuntu/ about installing the lighttpd server.

3. Install and enable the PHP modules

To run the webpages we need to have php modules installed. If you have already installed them then you might need to check if the modules are enabled.

Install PHP modules

```
sudo apt install php php-cgi
```

10. Create the mk14 directory and give the webservice update access to it

Now to enable the PHP module, run the following commands in the terminal.

```
sudo lighty-enable-mod fastcgi
sudo lighty-enable-mod fastcgi-php
```

After enabling modules, reload the Lighttpd server configuration by running the below command.

```
sudo service lighttpd force-reload
```

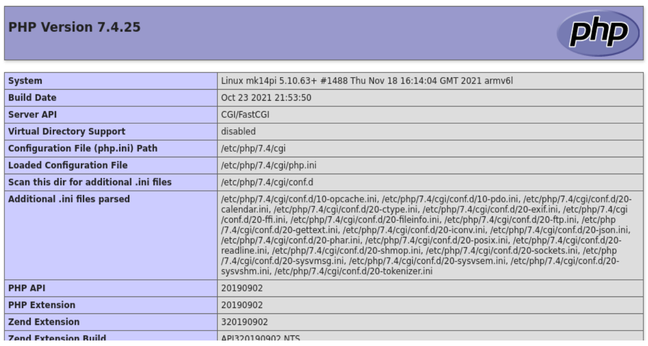
You can test if PHP is working or not, by creating a '**test.php**' file in **/var/www/test.php**.

```
sudo nano /var/www/html/test.php
```

Which contains

```
<?php phpinfo(); ?>
```

then go to http://server/test.php
and see a page similar to



PHP Version 7.4.25

System	Linux mk14pi 5.10.63+ #1488 Thu Nov 18 16:14:04 GMT 2021 armv6l
Build Date	Oct 23 2021 21:53:50
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/cgi
Loaded Configuration File	/etc/php/7.4/cgi/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/cgi/conf.d
Additional .ini files parsed	/etc/php/7.4/cgi/conf.d/10-opcache.ini, /etc/php/7.4/cgi/conf.d/10-pdo.ini, /etc/php/7.4/cgi/conf.d/20-calendar.ini, /etc/php/7.4/cgi/conf.d/20-ctype.ini, /etc/php/7.4/cgi/conf.d/20-exif.ini, /etc/php/7.4/cgi/conf.d/20-ffi.ini, /etc/php/7.4/cgi/conf.d/20-fileinfo.ini, /etc/php/7.4/cgi/conf.d/20-ftp.ini, /etc/php/7.4/cgi/conf.d/20-gettext.ini, /etc/php/7.4/cgi/conf.d/20-iconv.ini, /etc/php/7.4/cgi/conf.d/20-json.ini, /etc/php/7.4/cgi/conf.d/20-phar.ini, /etc/php/7.4/cgi/conf.d/20-posix.ini, /etc/php/7.4/cgi/conf.d/20-readline.ini, /etc/php/7.4/cgi/conf.d/20-shmop.ini, /etc/php/7.4/cgi/conf.d/20-sockets.ini, /etc/php/7.4/cgi/conf.d/20-sysvmsg.ini, /etc/php/7.4/cgi/conf.d/20-sysvsem.ini, /etc/php/7.4/cgi/conf.d/20-sysvshm.ini, /etc/php/7.4/cgi/conf.d/20-tokenizer.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,NTS

Note to see the webserver error log then use

```
tail /var/log/lighttpd/error.log
```

4. Configure the "php.ini" File

First, ensure that PHP is configured to allow file uploads.
You might find that it is already set to allow uploads.

This is in the "php.ini" file, which is shown on the test.php page above as

```
/etc/php/7.4/cgi/php.ini
```

Edit the file

```
sudo nano /etc/php/7.4/cgi/php.ini
```

and search for the `file_uploads` directive, and set it to On:

```
file_uploads = On
```

save the file and reload the Lighttpd server configuration by running the below command.

```
sudo service lighttpd force-reload
```

5. Configure the mimetype on the webserver

We need to tell the web server to treat .hex files as text files, if we don’t do this then when we try and show the contents of the programs on the web page it will ask where you want to download them to.

The default server comes with some scripts which generate the mime config setting from the base operating system config file. I decided it was easier to edit the /etc/lighttpd/lighttpd.conf file to handle the mime types as I wanted.

Edit the config file

```
sudo nano /etc/lighttpd/lighttpd.conf
```

comment out

```
#include_shell "/usr/share/lighttpd/use-ipv6.pl " + server.port
#include_shell "/usr/share/lighttpd/create-mime.conf.pl"
```

and then add these lines

```
mimetype.assign = (
    ".ico"   => "image/x-icon",
    ".jpeg"  => "image/jpeg",
    ".jpg"   => "image/jpeg",
    ".png"   => "image/png",
    ".svg"   => "image/svg+xml",
    ".css"   => "text/css; charset=utf-8",
    ".html"  => "text/html; charset=utf-8",
    ".js"    => "text/javascript; charset=utf-8",
    ".json"  => "application/json; charset=utf-8",
    ".map"   => "application/json; charset=utf-8",
    ".txt"   => "text/plain; charset=utf-8",
    ".hex"   => "text/plain; charset=utf-8",
)
```

save the file and restart the web server

```
sudo service lighttpd force-reload
```

6. Copy the MK14keys files to the webserver directory

The files in the html folder, and in the subdirectory called scripts, need to be copied to the base directory of the webserver.

Copy these files using an sftp client like filezilla to a html folder in the pi home path.

Then copy the files to the base directory of the webserver (/var/www/html).

```
sudo cp -R html/* /var/www/html/
```

You can check they all copied across by doing a ls

```
ls /var/www/html
```

The structure under the /var/www/html directory should look like this

```
├── action.php
├── delete.php
├── favicon.ico
├── format.php
├── formatyes.php
├── getosver.php
├── index.lighttpd.html
├── index.php
├── inputchars.php
├── program.php
├── restorefile.php
├── restorefiles.php
├── scripts
│   ├── send14_file.py
│   └── send14_string.py
├── sendchars.php
├── sendfile.php
├── setosver.php
├── test.php
├── uploadfile.php
└── upload.php
```

7. Create the mk14 directory

To store the .hex files we need to create a mk14 sub directory and give the web service user (www-data) update access to it. For now I have given everyone access to the folder.

create mk14 folder and give all rw

```
sudo mkdir /var/www/html/mk14
sudo chmod -R go+rw /var/www/html/mk14
```

8. Allow the webserver to access the GPIO pins

To access the GPIO pins from the website we need to give the webserver user (www-data) permissions. This can be done by added www-data to the gpio group.

```
sudo usermod -a -G gpio www-data
```

A service restart may allow access

```
sudo service lighttpd force-reload
```

But I found it gave me this error message when trying to send characters from http://mk14pi/inputchars.php

```
Last keys sent: ERROR - No access to /dev/mem. Try running as root!
```

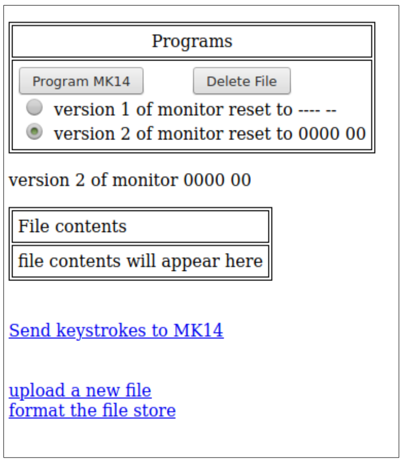
A reboot of the raspberry pi seemed to be needed to allow it to work.

9. Test the process.

From a web server call

```
http://mk14pi
```

and you should get the page



Programs

( ) version 1 of monitor reset to ---- --
[✓ version 2 of monitor reset to 0000 00]

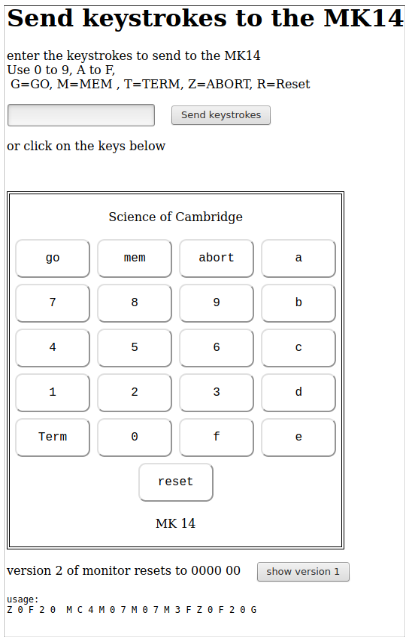
version 2 of monitor 0000 00

File contents
file contents will appear here

Send keystrokes to MK14

upload a new file
format the file store

select Send keystrokes to MK14 and you should see



Send keystrokes to the MK14

enter the keystrokes to send to the MK14
Use 0 to 9, A to F,
G=GO, M=MEM , T=TERM, Z=ABORT, R=Reset

or click on the keys below

version 2 of monitor resets to 0000 00

```
usage:
Z 0 F 2 0  M C 4 M 0 7 M 0 7 M 3 F Z 0 F 2 0 G
```

Connect the MK14 and try sending “keys” by pressing the buttons, or typing in the text box and pressing Send keystrokes

If you get the error

```
Last keys sent: ERROR - No access to
/dev/mem. Try running as root!
```

Then try rebooting the raspberry pi

If any of the pages are not displayed then check the web server log

```
tail /var/log/lighttpd/error.log
```