# Raspberry PI as the Supervisory Controller in Project VI (Elevator)

The steps already taken to set up the Raspberry Pi are outlined in this document. The SD cards in each elevator have all been imaged from a single SD card that has been set up using the process below. The process is outlined below in case modifications are required and for debug purposes.

If there are any problems with the SD cards in the elevators the first step would be to restore the image from the saved ISO using a Linux program called ‘Disks’. To create a new SD card simply format a new card and use Disks to ‘Restore an Image from a file’. There are a number of imaged SD cards available.

# Setting up a Raspberry Pi for this Project

1. USB/PCAN driver installation
2. Install make, gcc and g++, libstdc++, libpopot-dev

$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo apt-get dist-upgrade

$ sudo apt-get install build-essential

$ sudo apt-get install libpopt-dev

1. Check version of installed compilers

$ gcc -v

$ g++ -v

$ make -v

1. Install git (optional)

$ sudo apt-get install git

1. Next install the ‘tree’ command (optional)

$ sudo apt-get install tree

$ sudo apt-get update

1. Install Linux headers

$ sudo apt-get install linux-headers-rpi

$ sudo reboot

1. Download the driver from: <http://www.peak-system.com/fileadmin/media/linux/index.htm>
2. Copy the compressed driver file from your Downloads folder to ~/src/usb-can/

$ cd /usr/src/

$ sudo mkdir usb-pcan

$ cd ~/Downloads

$ sudo cp peak-linux-driver-8.5.1.tar.gz /usr/src/usb-pcan

1. Untar (unzip) the compressed driver file from your Downloads folder to ~/src/usb-can/

$ cd /usr/src/usb-pcan

$ ls - la

$ sudo tar -xzf peak-linux-driver-8.5.1.tar.gz

$ cd peak-linux-driver-8.5.1

1. Clean previous executables from the folder

$ sudo make clean

1. Build the libraries: On the Raspberry Pi there is no PCI so make using the following command!!!

$ sudo make PCI=NO\_PCI\_SUPPORT

1. Install the driver, user libraries and test programs

Be sure you are in the driver package root directory

$ cd peak-linux-driver-8.5.1

Install everything (root privileges required)

$ sudo make install

$ sudo reboot

1. Configure the default bit rate by changing the btr0btr1 value to 125 kb/s by doing the following:

* Install nano text editor

$ sudo apt-get install nano

* Modify the following file using nano

$ sudo nano /etc/modprobe.d/pcan.conf

* Modify the file to set the default bitrate via the btr0btr1 hexadecimal value to 125 kbps

The file should look like:

# pcan – automatic made entry, begin --------

# if required add options and remove comment

options pcan **btr0btr1=0x031C**  🡨 uncomment this line and replace with

install pcan /sbin/modprobe –ignore-install pcan

# pcan – automatic made entry, end -------

* Exit and save the file

1. Use the driver

The driver is automatically loaded for the target system (reboot the system) when the USB/CAN adapter is plugged in (Plug and play adapter). To load the driver manually, plug in the USB/CAN adapter and type:

$ sudo modprobe pcan

1. You can test the system and see that the bitrate has been properly set using

$ cat /proc/pcan

You will see a table of all the devices detected by the CAN driver

\*n -type -ndev- --base—irq –btr- --read-- --write-- --irqs-- -errors- status

**32** usb -NA ffffffff 02 **0x031C** 0000 00000 00000 0000 0x0000

1. Type

$ dmesg | grep pcan

And note the minor number for the USB (should be **32** to match the number from the /proc/pcan)

1. To see a list of the CAN interfaces use the tree command

$ tree -a /sys/class/pcan

You will see: **pcanusb32** -> ../../devices/virtual/pcan/pcanusb32

1. To get information about the PCAN interface type

$ lspcan -t –T -i

*Bus should say ACTIVE*

*Btrs = 125k*

*Port CAN1*

*pcanusb32*

## Install LAMPP on Linux (installs all modules needed)

1. Instal LAMPP from the command line

$ sudo apt-get install lamp-server^

1. Configure the Raspberry PI

$ sudo raspi-config

* Change user password: Use ‘ese’ as password
* Interfacing options: enable SSH and other capabilities as needed

1. You can install LAMP packages individually using

$ sudo apt-get update

$ sudo apt-get upgrade -y (the -y installs recommended pkgs as well)

$ sudo apt-get install apache2 -y

$ sudo a2enmod rewrite (enables modification)

1. Modify Apache to allow .htaccess overrides in the /var/www directory

$ sudo nano /etc/apache2/apache2.conf

Change ‘AllowOverride None’ to to ‘AllowOverride All’ as shown below

<Directory /var/www/>

Options Indexes

FollowSymLinks

AllowOverride All

Require all granted

</Directory>

Now if you place a file called .htaccess in the folder you can control the access to various directories. This is useful if you want to have multiple websites served from the same server.

Restart Apache

$ sudo service apache2 restart

1. Find the IP address for the PI and write it down

$ ifconfig

1. Install PHP (will install latest version – current PHP7)

$ sudo apt-get install php libapache2-mod-php –y

1. Test the PHP to see if it is working

$ cd /var/www/html

$ sudo nano index.php

Add the following code: <?php echo “Hello world”; ?>

Remove index.html

$ sudo rm index.html

Restart Apache

$ sudo service apache2 restart

1. Install mysql-server

$ sudo apt-get install mysql-server php-mysql -y

$ sudo service apache2 restart

1. Install mysql client

$ sudo apt-get install mysql-client

$ sudo apt-get install libmysqlclient-dev (this installs the client and mysql.h which is needed)

1. Install phpMyAdmin

$ sudo apt-get install phpMyAdmin –y

During installation choose:

* Apache2
* Click ‘yes’ to dbconfig-common
* Use the password ‘ese’
* The default username is ‘phpmyadmin’

Allow apache server to log in to phpMyAdmin

$ sudo nano /etc/apache2/apache2.conf

Edit the file by adding the following line at the end of the file:

Include /etc/phpMyAdmin/apache.conf

Restart the server

$ sudo service apache2 restart OR $ sudo /etc/init.d/apache2 restart

To get to phpMyAdmin open a browser and type

localhost/phpMyAdmin

The default user is ‘phpmyadmin’ the default password is ‘ese’

1. Test mysql client

$ sudo mysql -u root - p -P 3306 (Root database – access to all databases)

$ sudo mysql -u phpMyAdmin -p -P3306 (phpMyAdmin database)

Password is ‘ese’ for both logins

NOTE:

Need to use sudo for first login to MariaDb database 🡪 Need to add a password for ‘root’ account since it does not get added

$ sudo mysql -u root

MariaDb> SET PASSWORD FOR ‘root’@’localhost’ = PASSWORD(‘ese’);

Now you can login using

$ mysql -u root

1. Test connection to database via PDO (PHP database objects)

IMPORTANT NOTE: Need to create a new user ‘ese’ with password ‘ese’ since ‘root’ account in now blocked and cannot be accessed via PDO. (See course slides for NEW USER creation process).

* Created the ‘elevator’ database accessible by the ‘ese’ user account
* Created the elevatorNetwork table within it to modify/update
* Created the index.php file in the /var/www/html folder using code from the Software course to access the database
* Use the built-in editor ‘Geany’ to edit the files
* Test the code by typing ‘localhost’ in a browser on the PI or by typing the IP address of the pi into a browser on another computer on the same network.

## Install Connector/C++ to connect to MySQL database from C/C++ code

1. Download and install Connector C++
2. Download the libboost libraries

$ sudo apt-get install libboost1.62-all-dev

1. Install Connector/C++

$ sudo apt-get install libmysqlcppconn-dev -y (-y is optional)

* Library files (.a and .so files) copied to: /usr/lib/arm-linux-gnueabihf/
  + The files are
    - Libmysqlcppconn.a
    - Libmysqlcppcon.so
* Header files copied to include directory to: /usr/include/cppconn
  + Include these headers using: #include <cppconn/headerFilename.h>
* Using the sample program on the Desktop of the RPi make the file and execute