

IOT KERETRENDSZEREK ÉS IPARI ALKALMAZÁSAIK



SmartComLab

Gyors prototípus építés II.
Raspberry Pi, Linux, Python

Raspberry Pi 3B image

The image is for a 4GB microSD card (hopefully class 10 or above).

Preconfigured:

- Arrowhead core framework pre-configured
- Git, Maven, JDK 1.8, MySQL (MariaDB)
- PhpMyAdmin: <http://ip/phpmyadmin>
- Devtools (MCEdit, iptables...)
- Mosquitto MQTT broker
- The Raspberry hosts a Wi-Fi network and tunnels Internet if connected to eth0

Wi-Fi access point details:

- WiFi SSID: Arrowhead-Raspi-IoT#, password: arrowhead
- Raspi is DHCP master on the 192.168.42.10-50 range
- Raspi is the 192.168.42.1

<https://learn.adafruit.com/setting-up-a-raspberry-pi-as-a-wifi-access-point/install-software>

Raspbian: Debian alapú Linux distro

- Csomagkezelő: apt-get

4 x ARM 1.6 GHz mag, 512 MB RAM, SD kártya

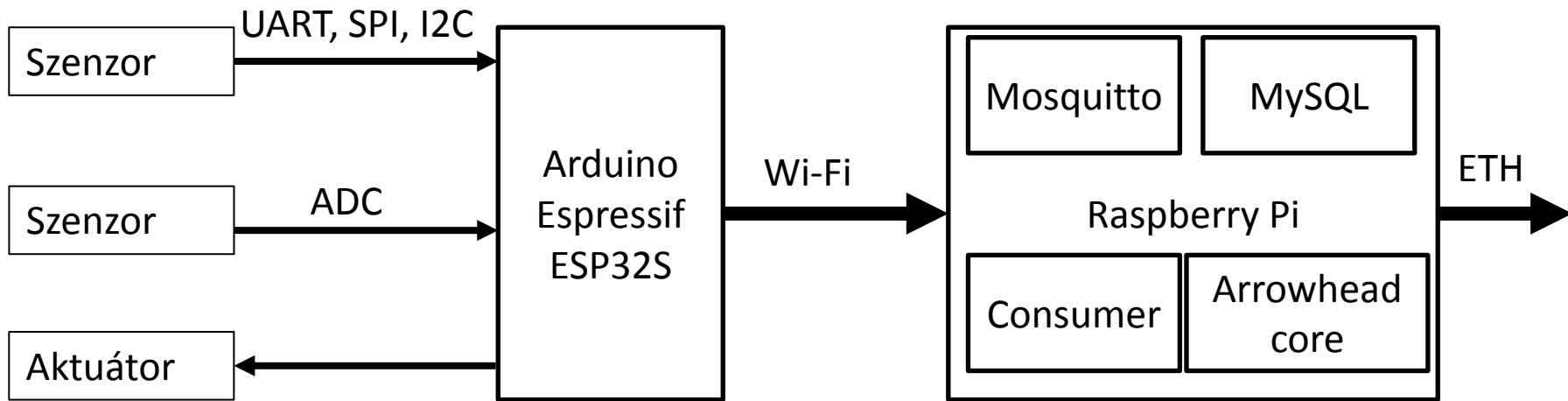


ARROWHEAD



SmartComLab

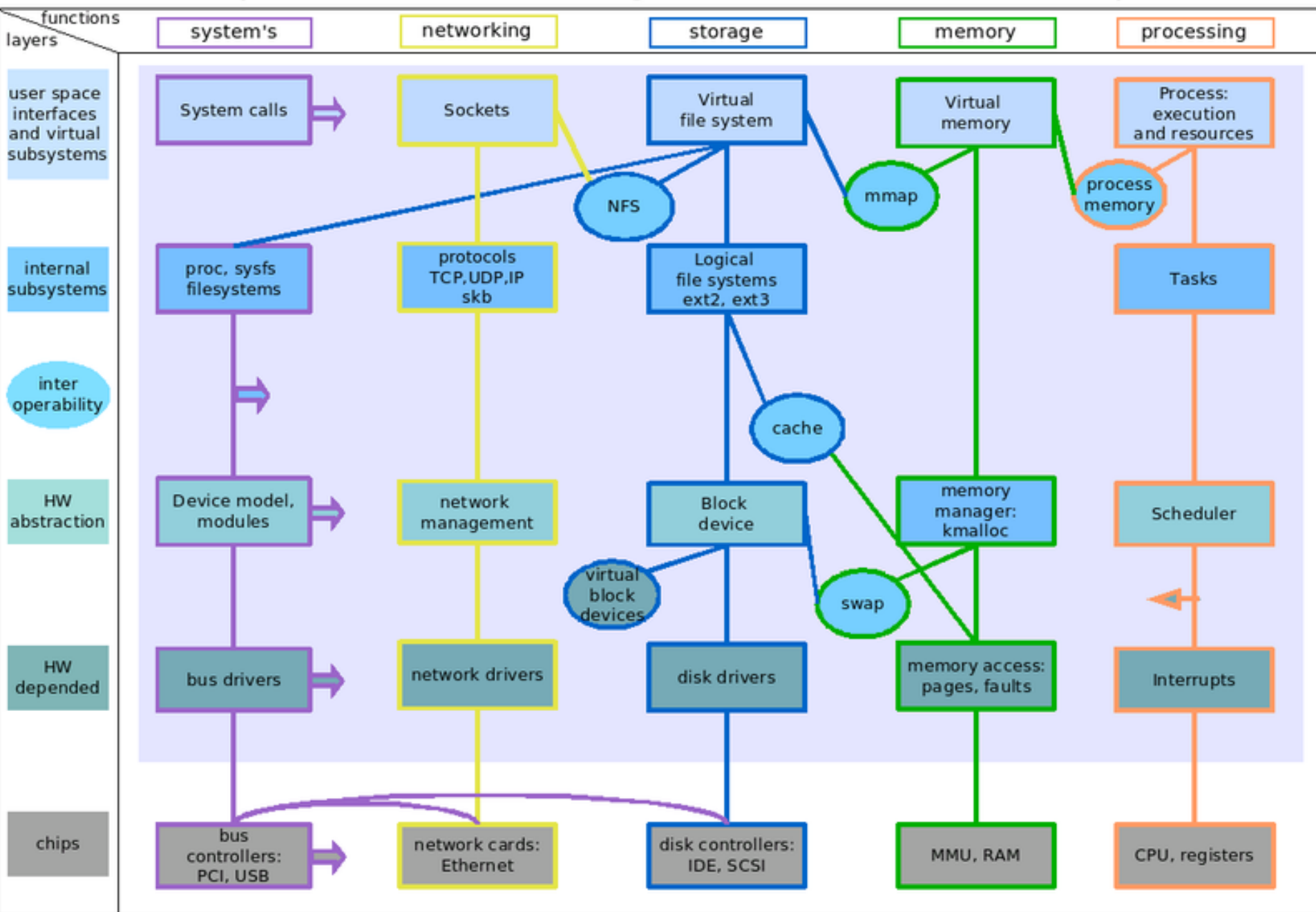
Házi feladat áttekintés



Tematika

- Linux bevezető
 - Könyvtárszerkezet
 - Alapvető bash parancsok
 - Futtatás indításkor, „service” létrehozása
- Python
 - Nyelv alapismerete
 - Néhány hasznos könyvtár

Simplified Linux kernel diagram in form of a matrix map



File System

- „device” vagy „partition” ami file-ok tárolására van formázva
- Sok formátum – a felhasználó előtt ugyanolyan megjelenés: könyvtárszerkezetben
- /proc/filesystems - pl. ext2 ext3 vfat NFS ntfs
- A hozzáféréshez *mount*-olni kell – így könyvtárként látszik/elérhető
 - mount -t vfat /dev/hda1 /windows/cdrive*
- mount / umount /eject

File System (cont)

- Applikáció is
 - Az adatahozzáférés vezérlésére
 - A könyvtárak kezelésére és hozzáféréséhez
 - Más applikációk is használják a szolgáltatásait
 - állományokat és könyvtárakat készít
 - megnyit és módosít létezőket
 - törlés
 - hozzáférés-szabályozás

Néhány főbb könyvtár

/bin	<i>common programs, shared by admins, users</i>
/boot	<i>startup files and kernel. Also GRUB data</i>
/dev	<i>references to all hardware as special files</i>
/etc	<i>important system configuration files. Sort of like Control Panel</i>
/home	<i>contains home directories for most users</i>
/initrd	<i>information contained for booting.. Don't mess with it!</i>
/lib	<i>library files, common files needed by many programs</i>
/lost+found _d	<i>files that were recovered after a system failure</i>
/mnt	<i>mount point for all external devices</i>
/net	<i>mount point for remote filesystems</i>
/opt	<i>extra third party software</i>
/proc	<i>info about system resources, man proc</i>
/root	<i>home dir. for the the admin</i>
/tmp	<i>temporary space</i>
/usr	<i>program, libraries, docs for most user programs</i>
/var	<i>other temporary files, such as logs, downloaded files, mail queue</i>

Alapvető műveletek

- `cd` könyvtárak közötti navigációhoz
- `mkdir` könyvtár létrehozása
- `rmdir` könyvtár törlése
- `ls` könyvtár tartalma
- `echo $PATH` végrehajtható parancsok helyei
pl.: /usr/local/bin:/usr/bin:/bin/
- `export` környezeti változó beállítása
pl. PATH=\$PATH:/new/directory/path
- `man` help a parancsok használatához
- `history` eddig használt parancsok listája
- `&` parancs után írva: fusson a háttérben *ls & -> [1] 23142*
- `fg` futó job visszahozása előtérbe *fg 23142*

Szöveges manipuláció

- cat teljes file szövegének összeállítása (kiírás)
- tac utolsó sor legelöl (hasznos pl. log file esetén)
- head a file kezdő sorai (default:10)
- tail a file záró sorai (default:10)
- more szöveg kiírása oldalanként a terminálra
- less szöveg kiírása: a felhasználó mászkálhat benne

regxp – regular expressions

- grepszövegben keresés szabályok szerint
- (g)awk szövegben keresés/manipuláció
- sed szöveg egyszeri futás alatti szerkesztése
- vi klasszik szövegszerkesztő
- emacs legendás szövegszerkesztő

Accounting

- Multiuser operációs rendszer
- Account:
 - felhasználónév azonosítja,
 - jelszó védi
- Kétféle account:
 - Root/superuser: bármit megtehet
 - User: saját userspace-én garázdálkodhat csak
- Sudo: superuser do

Accounting (cont)

- `su` root átvált az adott **userre/ré**
- `Sudo` root jogokkal futtatni egy adott parancsot
- `adduser` felhasználó hozzáadása
- `passwd` jelszó megváltoztatása
- `userdel` felhasználói account törlése
- `/etc/passwd` felhasználói account információk
- `/etc/shadow` felhasználói jelszó titkosítva; valamint account- és jelszó-lejáratati infók

Authorization: ownership & access

- Hozzáférés
 - Multi-user környezet!
 - Felhasználók nem férnek hozzá egymás file-jaihoz
 - Speciális file-okhoz csak a root fér hozzá
- Csoportok – groups
 - a felhasználók több csoporthoz tartozhatnak
 - könnyebb/rugalmasabb hozzáférés-kezelés
- Hozzáférés: felhasználók / csoportok / egyéb

Hozzáférés - példa

- „ls -l” - parancs példa kimenete

```
-rw-r--r--  1  raheel  raheel 32873 2006-2-04 2:24 new.txt
```

permissions no. of items, if directory user group size date and time last accessed name

```
-rw-r--r--
```

simple file user can read and write group can read only others can read only

Hozzáférés változtatás

- *chmod* parancs
- *r, w, x: read, write, execute*
- *Root: megváltoztathatja bármely file/directory hozzáférés-szabályait*
- *A root mellett csak a birtokos (user) tudja változtatni*

pl. chmod 755 myfile

Ownership változtatás

- Ownership (birtoklás) változtatás: `chown`
chown username file_or_dir
- Csoport-birtoklás változtatás: `chgrp`
chgrp groupname file_or_dir
- Kombinált csoport és felhasználónév:
chgrp name.name file_or_dir

Automatikus indítás

- „Service” létrehozásával a kódod automatikusan fog elindulni boot után
- Tutorial:

<https://www.raspberrypi.org/documentation/linux/usage/systemd.md>

Python 3 bevezető

Áttekintés

- Miért épp a Python?
- Python referencia & tutorial @ python.org
 - <https://docs.python.org/3/tutorial/>
 - <https://docs.python.org/3/reference/index.html>
- Python áttekintés: Cheat Sheets
- Gyakorlatok
- Best Practices
 - Általában a szkripteléshez
 - Python-hoz - <https://docs.python-guide.org/>
- Python Standard Library - <https://docs.python.org/3/library/>

Miért épp a Piton?

- Általános célú nyelv, amin könnyű írni és olvasni
- Nagyon sok lehetőséget kínál, különféle területeken vannak kiváló „library”-jai
 - Machine Learning
 - Big Data
 - Data Science
 - Web Design
 - System Administration
- Egészséges, aktív közösség építi, szépen fejlődik
- Nagyvállalatok (is) szponzorálják

Python referenciák, tutorial-ok

- <https://www.sololearn.com/Course/Python/>
- <https://docs.python.org/3/tutorial/>
- <https://developers.google.com/edu/python/>
- <https://docs.python.org/3/reference/index.html>

Cheat Sheets

- https://perso.limsi.fr/pointal/_media/python:cours:mementopython3-english.pdf
- <http://www.webpages.uidaho.edu/~stevel/504/Python%20Notes.pdf>
- https://ugoproto.github.io/ugo_py_doc/Python-Cheat-Sheet-by-CodeConquestDOTcom.pdf
- https://github.com/ehmatthes/pcc/releases/download/v1.0.0/beginners_python_cheat_sheet_pcc_all.pdf
- http://coffeeghost.net/pybat/python_cheatsheet.png
- <https://docs.python-guide.org/> → Hitchhiker's Guide to Python

Quick Python Script Explanation for Programmers

Load other code modules.

Module name. This refers to "os.py"

The name "main" is just a convention, not a requirement. See the very bottom of this script.

Newline automatically added to print statements. Also, there are no semicolons at the end of the line.

I prefer single-quotes, but either is fine. Either way, you don't have to escape the other kind of quote inside the string.

Function call.

String replication. Evaluates to '====='

String concatenation.

Call a function in the os module.

Variables MUST be instantiated first.

Lists can contain different data types in the same list, including other lists.

For loop. "i" takes on each value in the list "food" in order.

The range() function returns a list like [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]. Don't forget the colon at the end!

String interpolation works basically the same way as it does in C.

The comparison operators are the same as C.

Boolean operators are words, not && and ||.

Comments.

Colons come after def, for, while, if, elif, and else statements.

Multi-line strings don't affect block indentation, though, only the indentation at the START of the statement or expression.

We put a call to main() at the bottom so that each def statement is executed by the time we call main(). This script's __name__ variable has the value '__main__' only when the script is run, not imported. With this check, the main() function won't run if another script imports this script.

One-line block. When the indentation goes back down, the block has ended.

Function definition. Don't forget the colon at the end.

End of indentation, not a } brace, signifies the end of the block.

```

import os
def main():
    print 'Hello world!'
    print "This is Alice's greeting."
    print 'This is Bob\'s greeting.'
    foo(5, 10)
    print '=' * 10
    print 'Current working directory is ' + os.getcwd()
    counter = 0
    counter += 1
    food = ['apples', 'oranges', 'cats']
    for i in food:
        print 'I like to eat ' + i
    print 'Count to ten:'
    for i in range(10):
        print i
def foo(param1, secondParam):
    res = param1 + secondParam
    print '%s plus %s is equal to %s' % (param1, secondParam, res)
    if res < 50:
        print 'foo'
    elif (res >= 50) and ((param1 == 42) or (secondParam == 24)):
        print 'bar'
    else:
        print 'moo'
    return res # This is a one-line comment.
    '''A multi-
line string, but can also be a multi-line comment.'''
if __name__ == '__main__':
    main()

```

Példakód

```
import pymysql.cursors
import paho.mqtt.client as mqtt
from datetime import datetime

print("debug1")

def onconnect(client, userdata, flags, rc):
    if rc==0:
        print("connected OK Returned code=", rc)
    else:
        print("Bad connection Returned code=", rc)

client = mqtt.Client("python1")
client.on_connect=onconnect
client.connect("192.168.1.148")
client.subscribe("esp/both")

print("debug2")

def onmessage (client, userdata, message):
    print("received msg;", str(message.payload.decode("utf-8")))
    print("topic:",message.topic)
    if message.topic == "esp/both":
        print("Saving to database")
        connection=pymysql.connect( host="localhost", user="root", password ="root", db="arrowhead", cursorclass=pymysql.cursors.DictCursor)
        try:
            with connection.cursor() as cursor:
                now = datetime.now()
                formDate = now.strftime ("%Y-%m-%d %H:%M:%S")
                sql = "INSERT INTO tempData (humidity, temperature, time) VALUES (" + str(message.payload.decode("utf-8"))+ ", '" + formDate$
                print(sql)
                cursor.execute(sql)
                connection.commit()
                print("MySQL commit successful.")
        finally:
            connection.close()
    print("MySQL connection closed.")
```