# Připojení a setup:

1. stáhnout nejnovější verzi Raspbianu ze stránek, zkopírovat na kartu (“sudo dd bs=4M if=path/to/image of=/dev/name-of-card”) - jmeno karty pomoci prikazu “lsblk”
2. kartu vložit do Raspi, připojit pomocí SSH (ssh [pi@x.x.x.x](mailto:pi@x.x.x.x) – defaultni heslo je raspberry)
3. (optional) – pro rychlý ssh login generovat keypair pomocí “ssh-keygen -t rsa” a následně nahrát do raspi pomocí ssh-copy-id [pi@x.x.x.x](mailto:pi@x.x.x.x).
4. (troubleshoot) – pro zjištění portu na kterém se nachazí raspi je možno použít příkaz “nmap -T4 -sP -v ipadress/mask (1.1.2.0/24)” – IP počítače pro dané rozhrání jde získat pomocí ifconfig.
5. Nastavení hesla pro raspi pomocí příkazu “passwd”
6. expand filesystému pomocí “sudo raspi-config”
7. (optional) nastavení ipv6 editací ipv6.conf kde na konec stačí připsat “blacklist ipv6” a restartovat (“sudo reboot”)
8. (optional) připojení k wifi
9. aktualizace software pomocí “sudo apt-get update && sudo apt-get upgrade”
10. instalace wiringPi pomoci serie prikazu:
    1. sudo apt-get install git
    2. git clone git://git.drogon.net/wiringPi
    3. cd wiringPi
    4. ./build
    5. vyzkouseni funkcnosti pomoci “gpio readall”
11. instalace a konfigurace netbeans:
    1. stahnout verzi netbeans pro C/C++
    2. pridat build target (Window | Services | C/C++ Build Hosts | Add New Host) , vybrat moznost synchronizace pracovnich slozek pomoci SFTP.
    3. (optional) je dobre prednastavit pristup pomoci ssh klice (ssh-keygen -t rsa + ssh-copy-id pi@x.x.x.x)
    4. pridat repozitar git (Team | Git | Clone – zadat informace) – [rychle info jak zacit zde](http://rogerdudler.github.io/git-guide/)
    5. naklonovat projekt z repozitare pomoci Git | Clone
    6. $$$ Profit $$$

# Cvičení 2

1. Blikání ledkou pomocí pwm

Definice funkce s parametry

*int* ***softwarePWM****(int pin, int period, unsigned int duty, unsigned int runtime, int error)*

|  |  |
| --- | --- |
| Argument | Popis |
| pin | Výběr výstupního pinu pro PWM modulaci |
| period | Perioda signálu v μsec |
| duty | Relativní šířka pulsu v % |
| runtime | Doba běhu v milisekundách, 0 pro nekonečno |
| error | Maximální přípustná chyba časování v procentech |

**Inicializace proměnných**

{

int unsigned start\_time = micros();

int unsigned start\_duration, start\_duration\_low;

pinMode(pin, OUTPUT);

if (duty > 100) {

duty = 100;

}

**Výpočet PWM modulace**

int duration\_high = (duty \* period) / 100;

int duration\_low = period - duration\_high;

**Hlavní Loop který počítá chybu časování a zapisuje PWM na definovaný PIN**

while (1) {

start\_duration = micros();

digitalWrite(pin, HIGH);

delayMicroseconds(duration\_high);

digitalWrite(pin, LOW);

delayMicroseconds(duration\_low);

unsigned int curr\_time = micros();

unsigned int real\_duration = (curr\_time - start\_duration);

// microsecs

int abs\_diff = real\_duration - (duration\_high + duration\_low);

// percent

int rel\_diff = 100 \* abs\_diff / period;

if (rel\_diff >= error) {

printf("%d - softwarePWM - maximální přípustná relativní chyba časování v procentech byla překročena", curr\_time);

} else {

printf("%d - softwarePWM - vše je OK !! :)", curr\_time);

}

printf("\n MaxDiff: %d%\n RelDiff: %d%\n AbsDiff: %d\n RealDur: %d\n", error, rel\_diff, abs\_diff, real\_duration);

fflush(stdout); /\* force it to go out \*/

if (runtime && micros() > (start\_time + runtime))

break;

}

}

|  |
| --- |
| Doporučení pro implementaci PWM:   1. Použít dedikovaných pinů (2 na raspi) pro PWM – použít knihovnu pigpio 2. Brát v potaz přesnost časovacích funkcí z různých zdrojů |

1. Vícevláknové programování a zpracování přerušení (doporučené čtení: [threading](http://wiringpi.com/reference/priority-interrupts-and-threads/), [wiringpi-Threading](http://wiringpi.com/reference/priority-interrupts-and-threads/))

Vypracovaná funkce neobsahuje všechny body zadání, proto vypíši způsoby realizace jednotlivých bodů:

* 1. Spuštění vlákna

Nejdříve je potřeba nadefinovat vlákno pomocí příkazu:

PI\_THREAD (myThread)

{ spustitelny kod }

Takto nadefinovaný Thread se pak spustí v hlavním programu pomocí příkazu:

x = piThreadCreate (myThread) ;

if (x != 0)

printf ("it didn't start");

* 1. Problematika vláken pracujících s jednou globální proměnnou

Pokud vlákna sdílejí jednu nebo více globálních proměnných, ať už pro čtení či psaní, je potřeba nějak ošetřit tuto proměnnout aby se nemohla změnit v průběhu čtění/psaní. Od toho je tu wiringpi funkce piLock/piUnlock(int KeyNum). KeyNum je číslo od 0 do 3.

Definice klíče je:

# DEFINE COUNT\_KEY 0 //nebo také 1/2/3

Uzamčení proměnné pomocí klíče pak proběhne pomocí:

piLock(COUNT\_KEY);

++globalvalue;

piUnlock(COUNT\_KEY);

Pro demonstraci využití lockingu se koukni na ***wfi.c*** ve složce wiringPi/examples

* 1. Přerušování vláken

Probíhá registrací handlovací funkce na daný pin pomocí příkazu

int wiringPiISR (int pin, int edgeType,  void (\*function)(void)) ;

Kde pin je číslo pinu který bude interrupt registrovat, edgeType je INT\_EDGE\_[FALLING | RISING | BOTH | SETUP] a poslední argument je ukazatel na handlovací funkci.

Tedy je potřeba:

deklarovat handlovací funkci:

void myInterrupt(void)

{

++globalvar;

}

a tuto funkci přidat k vybranému pinu

wiringPiISR(Pin\_num, INT\_EDGE\_RISING, &myInterrupt)

Pro demonstraci přerušování vláken se koukni na ***isr.c*** ve složce wiringPi/examples

**POZOR NA ZKRATOVÁNÍ VÝSTUPNÍCH PINŮ!**

1. Připojení I2C zařízení
   1. Instalace potřebných nástrojů

nastavíme I2C na raspi, pomocí příkazu:

sudo raspi-config → advanced options → I2C enable

Následně z repo sehnat:

sudo apt-get install i2c-tools