ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

ФИЛИАЛ ПЛОВДИВ

ФАКУЛТЕТ ПО ЕЛЕКТРОНИКА И АВТОМАТИКА

КАТЕДРА “КОМПЮТЪРНИ СИСТЕМИ”

Разпределени вградени системи

Курсова работа

на

Иван Здравков – 611328

Трифон Дарджонов – 611316

Велизар Минчев - 611378

Тема:

<u>Система за следене на състоянието на мрежова апаратура</u>

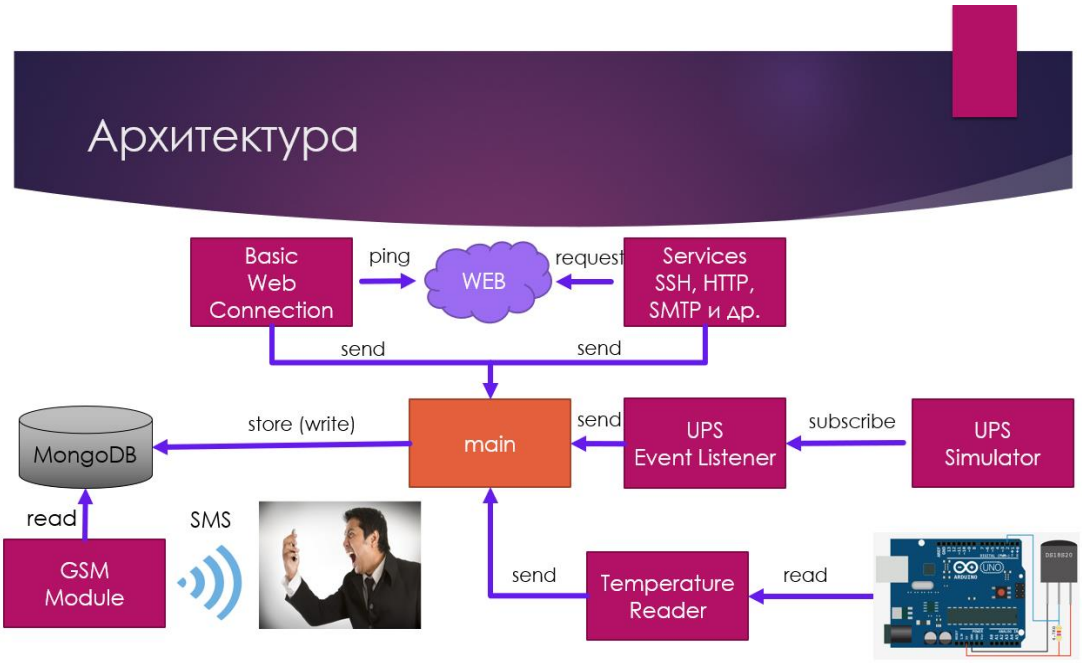
Приел: .....

/доц. Н. Киканаков/

Задание

Системата да следи следните параметри: отпадане на захранващото напрежение + UPS, температура, мрежова свързаност на възлите (ping), работа на услуги (ssh, http, smtp и др) по зададен адрес, порт и схема (протоколо).

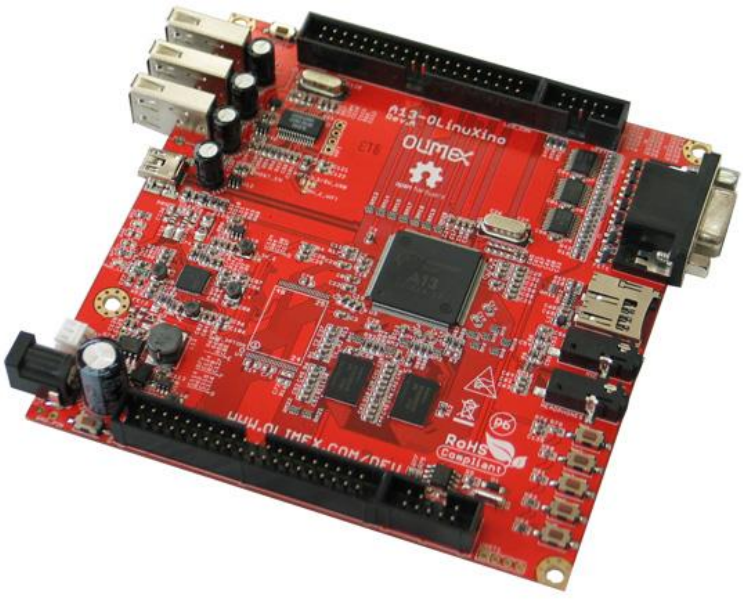
При отпадане на услуга или излизане на следените параметри от определени граници да изпраща SMS и да може да рестартира мрежов възел при команда през GSM.



Хардуер

OLinuXino A13

- A13 Cortex A8 processor at 1GHz, 3D Mali400 GPU
- UEXT: MOD-TC
- USB: USB-ETHERNET-AX88772B
- Симулиране на UPS
- UEXT/GPIO: PIG-GSM



Разпределение на задачите

Трифон: Модул за следене на отпадане на захванващото напрежение + UPS. Мрежова свързаност на възлите (ping)

Велизар: Модул за следене на температурата. Linux администрация и администриране на база данни.

Иван: SSH, HTTP, SMTP и др. по зададен адрес, порт и схема(протокол), връзка с базата данни /файл за съхранение на информацията/

Source Contol

https://bitbucket.org/DistributedEmbeddedSystemsTeam/network-monitoring-system

config.ini

За конфигуриране на променливите при работата на приложението, използваме config file

HttpServices.py

Входната точка на приложението, инициализира всички останали модули, HTTP Server-а и Listener-те. При промяна на състоянието на системата, записва в MongoDB базата данни или файловата система. При HTTP заявка от клиента на релативен път, сървъра връща текущото състояние на системата.

Този class се грижи за това, какво се случва с данните при извикване на някой от callback функциите. Той се грижи и за това къде да бъдат записани резултатите, като за целта имаме generic repository което се инициализира в конструктура на класа. В момента ползваме FileRepo (self.repo = FileRepo()), но не пречи да бъде сменено с друга имплементация (например MongoDB). Този class реално съдържа business rules на системата, и него inject-ваме в отделните имплементации на отделните класове. С това се стремим да поставим ясно разделение между low level details като четене на данни, и high level policy-та. За целите на проекта не сме го разбили на по малки class-ве и не inject-ваме репозитори-то през конструктура както и други добри практики които умишлено нарушаваме.

Routes:

/getTemperature – The current temperature is X

/getUPSStatus – The UPS is currently: ON/OFF

/isThereInternet - The internet connection is ON/OFF

SMTP.py

SMTP модула изпраща Email-и, използвайки SMTP клиент

UPSListener.py

Приема конкретна имплементация на UPS, като в момента имплементацията е симулация на UPS.

{

'isPowerDown': (random.randint(0, 1) == 0),

'batteryLifeTime': 30,

'batteryFullLifeTime': 120

}

В зависимост от това дали захранването минава в state Up или Down се извикват различни callback функции които получават следния обект като параметър

{

'timeElapsed': self.inSeconds(self.endTime - self.startTime), //timespan от предната промяна

'batteryLifeTime': UPSResponse['batteryLifeTime'],

'batteryFullLifeTime': UPSResponse['batteryFullLifeTime']

}

NetworkConnectivity.py

Този модул проверява за layer 3 свързаност (ping) към подаден IP адрес.

{ 'timeElapsed': X, 'host': host }

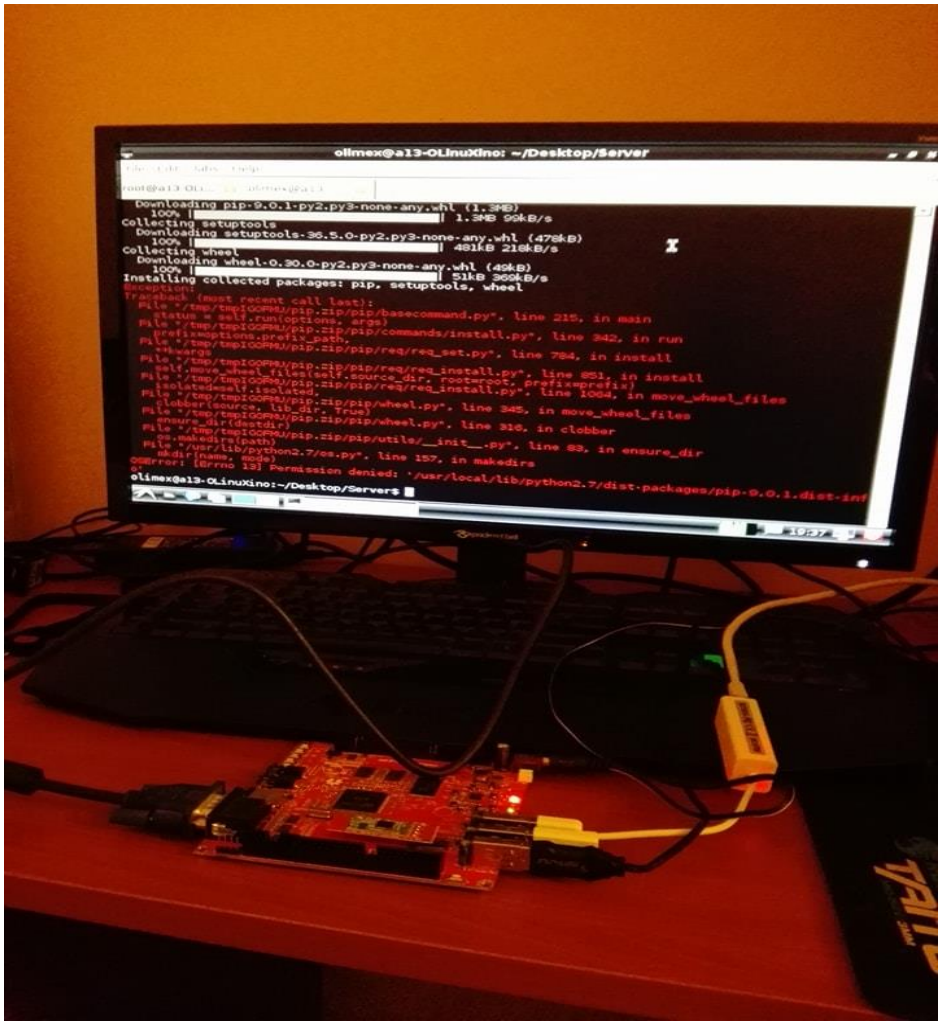
Това е модела който се подава съответно на invoke-натия callback onPingPass/onPingFail.

FileRepo.py

Конкретна имплементация със съхранение в файл на нужните ни по задание функционалности (добавяне на нов запис, и взимане на последния запис). С учебна цел не сме спазвали best practices.

MongoDBRepo.py

Поради ограниченията на хардуера на платката, MongoDB не може да бъде инсталирано, поради което сме подготвили MongoDBRepo, което запазва същата структира, като FileRepo-то. При отстраняване на техническите неизправности, имплементацията на FileRepo, може да бъде подменена с MongoDBRepo, без никакви други промени по съществуващите файлове, освен кой модул се инициализира.

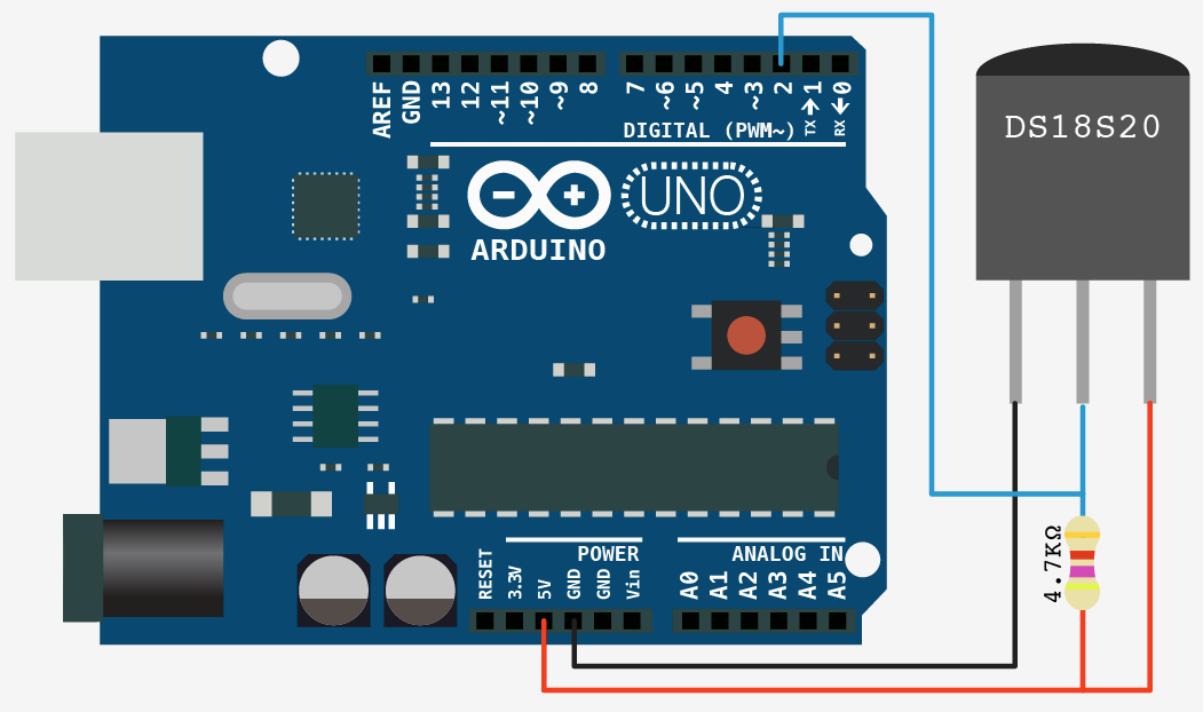


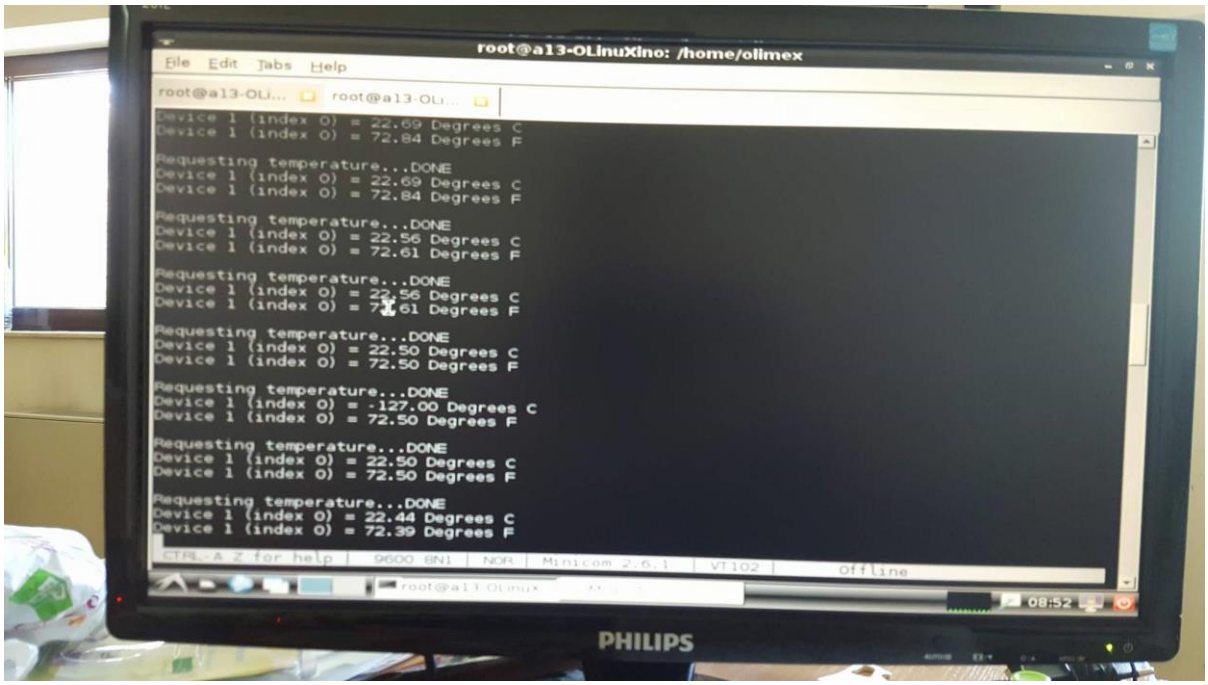
DS18B20.c - температурен сензор

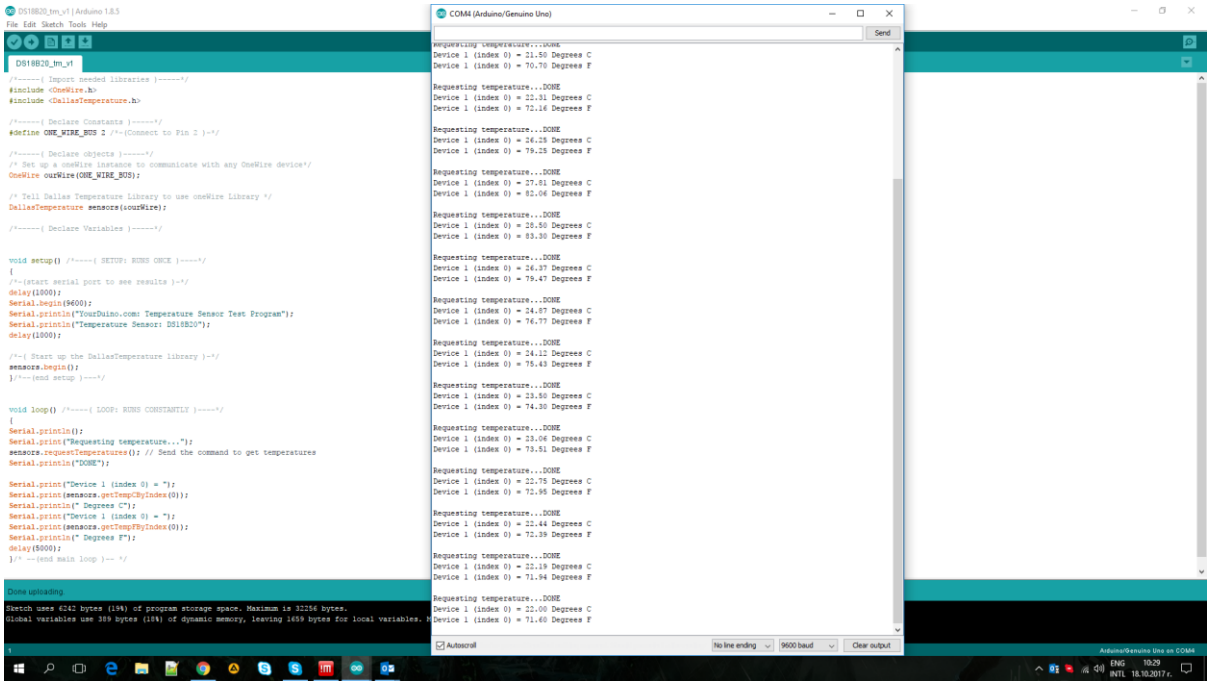
За да реализирам отчитане на температура използвах температурен сензор DS18B20 свързан към платка Arduino Uno, което е захранвано през USB порт от OlinuXino A13(Чрез USB порта осъществявам и четенето на данните от термометъра). През Arduino Uno се осъществява четене от сензора през интервал 9600 BR(baud rate). Четенето става през сериен порт. След това стартирам приложението minicom то се настройва от кой порт да чете и през какво време се изпълнява четенето на температурата. Със следващата команда пускам постоянен поток, който следя :

minicom –D /dev/ttyACM0 –C temp_status

Първоначално се опитах да реализирам отчинатането на температурата чрез периферно устройство MOD-TC. След това реших да го реализирам чрез дирепкно свързване през GPIO-1 пиновете на OlinuXino A13, но поради проблеми с драйверите и с библиотеките за One Wire протокол се наложи използването и на друго периферно устройство. За това реализирах четенето от датчика чрез допълнилна платка(Arduino Uno).


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
ARDUINO
UNO
DS18S20
4.7KΩ
RESET
3.3V
5V
GND
GND
Vin
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN


root@a13-OLinuXino: /home/olimex
File Edit Tabs Help
Device 1 (index 0) = 22.69 Degrees C
Device 1 (index 0) = 72.84 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.69 Degrees C
Device 1 (index 0) = 72.84 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.56 Degrees C
Device 1 (index 0) = 72.61 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.56 Degrees C
Device 1 (index 0) = 72.61 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.50 Degrees C
Device 1 (index 0) = 72.50 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = -127.00 Degrees C
Device 1 (index 0) = 72.50 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.50 Degrees C
Device 1 (index 0) = 72.50 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.44 Degrees C
Device 1 (index 0) = 72.39 Degrees F
CTRL-A Z for help | 9600 8N1 | NOR | Minicom 2.6.1 | VT102 | Offline
08:52
PHILIPS


DS18B20_tm_v1 | Arduino 1.8.5
File Edit Sketch Tools Help
DS18B20_tm_v1
/*-----(Import needed libraries)-----*/
#include <OneWire.h>
#include <DallasTemperature.h>
/*-----(Declare Constants)-----*/
#define ONE_WIRE_BUS 2 /*-(Connect to Pin 2)-*/
/*-----(Declare objects)-----*/
/* Set up a oneWire instance to communicate with any OneWire device*/
OneWire ourWire(ONE_WIRE_BUS);
/* Tell Dallas Temperature Library to use oneWire Library */
DallasTemperature sensors(&ourWire);
/*-----(Declare Variables)-----*/
void setup() /*----(SETUP: RUNS ONCE)----*/
{
/*-(start serial port to see results)-*/
delay(1000);
Serial.begin(9600);
Serial.println("YourDuino.com: Temperature Sensor Test Program");
Serial.println("Temperature Sensor: DS18B20");
delay(1000);
/*-(Start up the DallasTemperature library)-*/
sensors.begin();
}/*--(end setup)---*/
void loop() /*----(LOOP: RUNS CONSTANTLY)----*/
{
Serial.println();
Serial.print("Requesting temperature...");
sensors.requestTemperatures(); // Send the command to get temperatures
Serial.println("DONE");
Serial.print("Device 1 (index 0) = ");
Serial.print(sensors.getTempCByIndex(0));
Serial.println(" Degrees C");
Serial.print("Device 1 (index 0) = ");
Serial.print(sensors.getTempFByIndex(0));
Serial.println(" Degrees F");
delay(5000);
}/* --(end main loop)-- */
Done uploading.
Sketch uses 4242 bytes (13%) of program storage space. Maximum is 32256 bytes.
Global variables use 389 bytes (19%) of dynamic memory, leaving 1659 bytes for local variables.
COM4 (Arduino/Genuino Uno)
Send
Requesting temperature...DONE
Device 1 (index 0) = 21.50 Degrees C
Device 1 (index 0) = 70.70 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.31 Degrees C
Device 1 (index 0) = 72.16 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 26.25 Degrees C
Device 1 (index 0) = 79.25 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 27.81 Degrees C
Device 1 (index 0) = 82.06 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 28.50 Degrees C
Device 1 (index 0) = 83.30 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 26.37 Degrees C
Device 1 (index 0) = 79.47 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 24.87 Degrees C
Device 1 (index 0) = 76.77 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 24.12 Degrees C
Device 1 (index 0) = 75.43 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 23.50 Degrees C
Device 1 (index 0) = 74.30 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 23.06 Degrees C
Device 1 (index 0) = 73.51 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.75 Degrees C
Device 1 (index 0) = 72.95 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.44 Degrees C
Device 1 (index 0) = 72.39 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.19 Degrees C
Device 1 (index 0) = 71.94 Degrees F
Requesting temperature...DONE
Device 1 (index 0) = 22.00 Degrees C
Device 1 (index 0) = 71.60 Degrees F
Autoscroll
No line ending
9600 baud
Clear output
Arduino/Genuino Uno on COM4
ENG
INTL
10:29
18.10.2017 г.


PHILIPS
soup