File List.

PI

/etc/lirc/lircd.conf – Remote mapping (currently at 38kHz)

/home/pi/modules/mainSys.c – main system for mqtt and lirc

Compile: gcc –o [outputfile] mainSys.c –l mosquitto

double getTemperature()

get temperature data from sensor

double getCurr()

get current data from sensor

void messageArrived(struct mosquitto \*m, void \*obj, const struct mosquitto\_message \*message)

classify message received by mosquitto subscriber

format: \*ABC where

A = {d, u, 0, 1, r}. d = decrease temperature, u = increase temperature, 0 = power off, 1 = power on, r = read sensor

B (if only A is r) = power status of the AC {0, 1}. 0 = off, 1 = on

C = {16 … 30} (temperature)

For each type of command,

1. Build LIRC response

execute system call of LIRC

format: irsend SEND\_ONCE MY\_REMOTE COMMAND

COMMAND: a proper and suitable keypress defined in lircd.conf

1. (If command is r) read sensor data
2. Build MQTT response

format: P;T;S;C;% where P = power (ON/OFF), T = temperature (float), S = set/target temperature (int), C = current (float)

int main(int argc, char \*\*argv)

initialize I2C bus for temperature reading, initialize mosquitto, connect to server, and subscribe to the specified topic. Contains a main message-waiting loop

/home/pi/modules/i2c/ads1115.c – current sensor

/home/pi/modules/i2c/si7021-2.c – temperature/humidity sensor [b hum+temp; h hum; t temp] (for testing)

float getHumidity(int device)

returns humidity level

float getTemperature(int device)

returns temperature in ºC

int main(int argc, char \*\*argv)

initialize i2c bus and device

(for debugging) printing humidity level and/or temperature

(for debugging) saving log data to a file

/home/pi/modules/libcurl/tesh.c – php testing

tbniot public\_html/add\_data.php – php GET for MySQL add data

tbniot public\_html/connect.php – php credential for MySQL

tbniot public\_html/index.php – php for showing sensor data

tbniot public\_html/ctrl.html – controller html

Showing AC status and providing AC control

Javascript main:

initializations, display infos, bind events

publish\_msg() (for debugging/tester)

composing a MQTT message and publishing it to the corresponding MQTT topic

publish(txt)

composing a MQTT message and publishing it to the corresponding MQTT topic

resetting the timer of connection checker

temp\_up()

sending command to increase temperature

temp\_down()

sending command to decrease temperature

read\_sensor()

sending command to get/refresh sensor reading

toggle\_power()

sending command to turn the AC’s power on/off and refresh the temperature reading

onConnectionLost(responseObject)

connection loss event

onMessageArrived(message)

message arrival event. Parsing the received message and updating the status fields

format: P;T;S;C;% where P = power (ON/OFF), T = temperature (float), S = set/target temperature (int), C = current (float)

send\_console(text)

add text to the debugging console

check\_response()

check whether all messages have been responded to determine if the systems (Pi and controller) are connected or not

/home/pi/CronJobs/ServerSend.sh

Automatic sensor sending every 5 minutes

/etc/cron.d/ServerSend

Location of cron job

Calls

How-To

1. Run mainSys
2. Open ctrl.html any time when mainSys is loaded

Credentials for web hosting:  
hosting -- 000webhost

hosting acc -- umumlar5@gmail.com

hosting pwd -- tbniot000

server -- tbniot.000webhostapp.com

server pw -- tbniot000

DB name -- id6526640\_table1

DB acc -- tbniot

DB pwd -- tbniot000

DB host -- localhost

Table name -- LOG

Table cols -- TIMESTAMP INT

CURRENT FLOAT

TEMPERATURE FLOAT