IOManager.cpp

// Created by damien on 08/10/23.

// modified by Patrick 12/10/2023, tout ok avec buton ou message MQTT

// modified by Patrick 14/02/2024 modif wiring of MCP23017, One for Aig et ond for return position

// modified by Patrick 30/03/2024 reading state of Aig position ton confirm right way

#include "IOManager.h"

#include "mqttManager.h"

#include <Adafruit\_MCP23X17.h>// add for I2C & MCP23017

#include <Wire.h> // add for I2C GPIO 5 (D1) (SCL), GPIO 4 (D2) (SDA)

constexpr uint64\_t pulseTime = 500;// 500 ms Constante pour la durée de l'impulsion

Adafruit\_MCP23X17 mcp1;// Adafruit\_MCP23X17 mcp1;

Adafruit\_MCP23X17 mcp2;// Adafruit\_MCP23X17 mcp2;

IOManager::IOManager() = default; // Constructeur par défaut pour IOManager

void IOManager::setup() {

switchState = false;// all Switch off

Wire.begin(4, 5);// SDA d2 GPIO4, SCL d1 GPIO5, esp8266...Start I2C communication

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Scanner I2C

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

byte error, address;

int nDevices;

Serial.println("Scanning...");

nDevices = 0;

for (address = 1; address < 127; address++) {

// The i2c\_scanner uses the return value of

// the Write.endTransmisstion to see if

// a device did acknowledge to the address.

Wire.beginTransmission(address);

error = Wire.endTransmission();

if (error == 0) {

Serial.print("I2C device found at address 0x");

if (address < 16)

Serial.print("0");

Serial.print(address, HEX);

Serial.println(" !");

nDevices++;

} else if (error == 4) {

Serial.print("Unknown error at address 0x");

if (address < 16)

Serial.print("0");

Serial.println(address, HEX);

}

}

if (nDevices == 0)

Serial.println("No I2C devices found\n");

else

Serial.println("done\n");

delay(200); // wait 0.2 seconds for next scan

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Init mcp1 output

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

mcp1.begin\_I2C(0x20); // Instantiate mcp1: objectaddress 0

if (!mcp1.begin\_I2C()) {

Serial.println(" Error.mcp1 "); // if (!mcp1.begin\_SPI(CS\_PIN)) { --> if use a spi version

while (1);

}

Serial.print(" mcp1 Ok");

Serial.println(" setting mcp1 output High (setup) ");

for (int n = 0; n < 16; n++) // configure Port A: pins 0..7/ Port B: pins 8..15

{

mcp1.pinMode(n, OUTPUT);

delay(50);

mcp1.digitalWrite(n, HIGH); // all setting off

}

delay(100);

Serial.println();

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Init mcp2 input & init position turnout

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

mcp2.begin\_I2C(0x21); // Instantiate mcp2: objectaddress 1

if (!mcp2.begin\_I2C(0x21)) {

Serial.println(" Error.mcp2 "); // if (!mcp2.begin\_SPI(CS\_PIN)) { --> if use a spi version

while (1);

}

Serial.print(" mcp2 Ok");

for (int i = 0; i < 16; i++) {

mcp2.pinMode(i, INPUT\_PULLUP); // init HIGH

delay(100);

}

byte input2[16];

Serial.print(" lecture position on mcp2 (setup): ");

for (int i = 0; i < 16; i++) {

input2[i] = mcp2.digitalRead(i);

if (i == 15) {

Serial.print(input2[i]);

Serial.print(" i == 15");

} else {

Serial.print(input2[i]);

if (i % 2 == 1) {

Serial.print(" ");

if (input2[i] != Contact[i]) { // int Contact[16] = { 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0};

Serial.print(" (aig ko: ");

Serial.print(PosBobine[i]);

Serial.print(") ");

}

}

if (i == 3 || i == 7 || i == 11) {

Serial.print(" ");

}

}

delay(5);

}

Serial.println("");

}

void IOManager::loop() {

if (timing >= 0) {

if ((millis() - pulseStop) >= pulseTime) {

setLEDState(timing, false, topic\_sub, "0");

return;

}

}

}

void IOManager::attachMqttManager(MqttManager \*mngr) {

mqttManager = mngr;

}

void IOManager::setLEDState(int8\_t Id, bool on, String topic\_sub, String Payload\_sub) {

if (on) { // digitalWrite(Id, LOW);// relais on, then off

timing = Id;

pulseStop = millis();

mcp1.digitalWrite(Id, LOW); // Relais on LOW

Serial.print(" --> mcp1 on for Id: ");

Serial.println(Id);

} else {

timing = -1;

mcp1.digitalWrite(Id, HIGH); // Relais off HIGH

Serial.print(" --> mcp1 off for Id: ");

Serial.println(Id);

delay(5);

byte input2[16];

Serial.print(" lecture position on mcp2: ");

for (int i = 0; i < 16; i++) {

input2[i] = mcp2.digitalRead(i);

if (i == 15) {

Serial.print(input2[i]);

} else {

Serial.print(input2[i]);

if (i % 2 == 1) {

Serial.print(" ");

}

if (i == 3 || i == 7 || i == 11) {

Serial.print(" ");

}

}

delay(5);

}

Serial.print(" --> Check position Aig for id ");

Serial.println(Id);

//const byte PosBobine[] = {11, 12, 21, 22, 31, 32, 33, 34, 41, 42, 51, 52, 61, 62, 63, 64};

String topic\_sub;

String Payload\_sub;

topic\_sub = "train/state/aig";

Payload\_sub = PosBobine[Id];

Serial.print("Payload\_sub: ");

Serial.print(Payload\_sub);

if (input2[Id] == 1) {

Serial.print(" topic: ");

Serial.print(topic\_sub);

Serial.print(" mess\_sub: ");

Serial.println(Payload\_sub);

String mess\_sub = Payload\_sub; // ajout pour state, voir à créer une variable state

mqttManager->senMessage(topic\_sub, Payload\_sub);

} else {

Serial.print(" wrong etat Aig: ");

Serial.println(input2[Id]);

}

}

}