TÉCNICO LISBOA

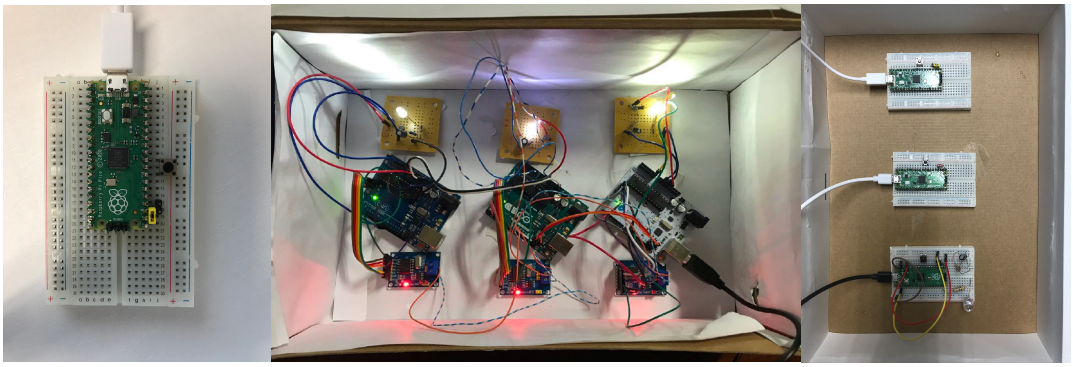
Master Degree in Electrical and Computer Engineering
2021/2022 – Spring Semester

Distributed Real-Time Control Systems
(Sistemas de Controlo Distribuído em Tempo-Real)

PART 2 – Group Project

(Last 3 lab sessions)

Real-Time Cooperative Decentralized Control of an Illumination Systems with a network of Raspberry Pi Pico microcontrollers.



Prepared by

Alexandre Bernardino, João Pedro Gomes, Ricardo Ribeiro

Instituto Superior Técnico

Department of Electrical and Computer Engineering

Scientific Area of Systems, Decision and Control

Version 1.0

Second Stage

1. Description

During the second stage the students will collaborate within a group to implement a cooperative distributed (i.e., non-centralized) control system with a PC-based interface.

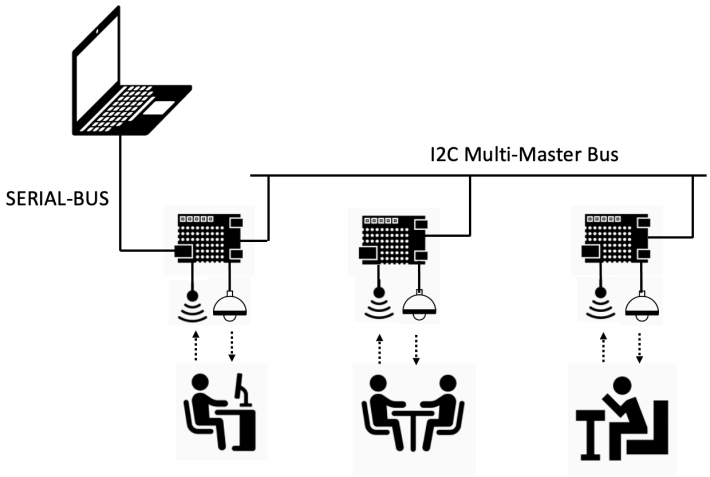


Figure 1 – Diagram of the distributed illumination network.

Three microcontroller nodes (luminaires) will be installed in the office model, one for each desk. The luminaires can communicate using a I2C bus network in Multi Master Mode – nodes can take initiative to initiate communications at any time. The luminaires should exchange messages whenever necessary to jointly minimize energy consumption and maximize user comfort. User comfort is defined as having a luminance on its desk always above or equal to the reference set points (visibility criteria) and low variations of high frequency noticeable to the human eye (flicker criteria). In this non-centralized cooperative distributed control mode, there is no central node. Ideally all luminaires should have the same code and implement "boot" mechanisms to recognize each other and be able to operate in a network with a variable number of nodes.

The PC interface will connect to one of the luminaires via USB serial communications. This luminaire will operate as a hub to interchange information between the network and the PC. The hub luminaire has two main purposes: (i) collect information from all control nodes (dimming levels, occupation levels, etc) and send it to the PC; (ii) receive commands from the PC (set points, working modes, etc) and route them the target control nodes.

2. System Functions

We can identify the following 4 main functionalities of the code to run in a control node: (i) inter-node communications; (ii) hub function; (iii) illumination system start-up and calibration; (iv) distributed control algorithm.

Inter-Node Communications

In cooperative distributed control systems, each node typically takes the initiative to send messages to its neighbours and is permanently listening for incoming messages from other nodes. Messages can be directed to any node, or they can be broadcasted to all nodes. Some messages can indicate a change of state that requires control actions by the neighbouring nodes, while others are for logging purposes.

Note 1: You should use the RPI pico C/C++ SDK function *flash_get_unique_id* to obtain a unique identifier for each node.

Note 2: Each i2c peripheral in the RPI pico cannot function simultaneously as master and slave. You should use one i2c channel as master (to send messages) and the other as slave (to receive messages).

Hub Function

A node, if connected to a PC, should activate hub function mode. This function adds, to the normal operation of a node, the role to communicate with the PC. The communication with PC is made with USB serial link, whereas the communication with the other nodes is made with the I2C bus. The hub node is responsible to route information from the other nodes to the PC and vice-versa.

Illumination System Calibration

When a luminaire changes its actuation, it impacts neighbouring luminaires (coupling effect). To realize the global controller, it is necessary to model the effects of the intensity of a luminaire in the measured illuminance of the other(s). Because this coupling between luminaires depends on many factors, a calibration procedure should be made at system start-up. For example, turn on one luminaire at each time and measure the illuminance in all other luminaires. This effect is almost linear when LUX units are used for the illuminance, and only a small number of measurements is required to model it. This method should also allow the estimation of the current external illumination.

Distributed Control Algorithm

The distributed controller should be “non-centralized”, i.e., no central master exists. This improves the overall reliability because the whole system can still operate even if one node stops working. Each node will be an independent decision-making unit with a

shared “goal”: to minimize a global cost function (energy consumption), while satisfying the minimal illuminance levels according to the desks’ occupation. Two cost functions will be considered: one with luminaires with identical power consumption, and another with luminaires with different powers. The coefficients of the cost function for each luminaire, which are proportional to its power consumption, can be set via the PC interface. A few distributed optimization algorithms will be given in the course lectures and the groups will have to choose which ones can be used and discuss the pros and cons of each alternative.

In addition to the list of commands defined in Part 1, the following commands, specific for Part 2, should be implemented:

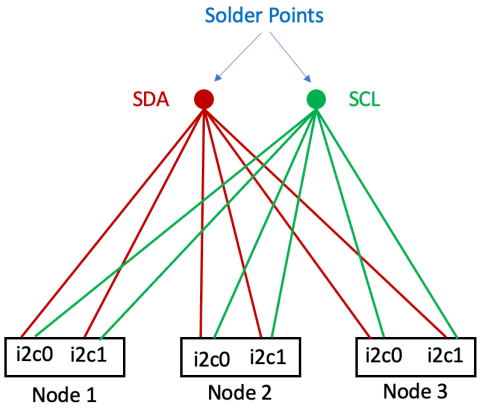
Command	**Client Request**	**Server Response**	**Observation**
Get lower bound on illuminance for Occupied state at desk <i>	“g O <i>”	“O <i> <val>”	<val> is a number expressing illuminance lower bound for Occupied state in desk <i> in lux.
Set lower bound on illuminance for Occupied state at desk <i>	“O <i> <val>”	“ack” or “err”	<val> is a number expressing illuminance lower bound for Occupied state in desk <i> in lux.
Get lower bound on illuminance for Unoccupied state at desk <i>	“g U <i>”	“U <i> <val>”	<val> is a number expressing illuminance lower bound for Unoccupied state in desk <i> in lux.
Set lower bound on illuminance for Unccupied state at desk <i>	“U <i> <val>”	“ack” or “err”	<val> is a number expressing illuminance lower bound for Unoccupied state in desk <i> in lux.
Get current illuminance lower bound at desk <i>	“g L <i>”	“L <i> <val>”	<val> is a number expressing illuminance lower bound in lux.
Get current energy cost at desk <i>	“g c <i>”	“c <i> <val>”	<val> is a number expressing the current cost of energy
Set current energy cost at desk <i>	“c <i> <val>”	“ack” or “err”	<val> is a number expressing the current cost of energy
Restart system	“r”	“ack” or “err”	Reset all values and recalibrate.

3. Cabling the I2C Bus

The I2C bus is not very robust to noise. It was designed to implement communication buses between integrated circuits in the same PCB or between PCBs with good connections. Its electric implementation is sensitive to the line capacities, that low-pass filter the signal and

reduce the communication rate. It is also sensitive to different signal delays in the bus lines due to different cable path lengths.

To reduce the effects describe above, we should avoid making too many electrical connections using jumper cables connected via the breadboard slots. It is recommended to make a special cable to connect the different microcontrollers. Each student has available in the lab monofilar cables to make two bunches of six cables each: one for the SDA and other for SCL (red and green bunches in the figure below). In each buch all cables should have the same length (about 15cm-20cm should be enough) and be twisted and soldered together in one of the ends. In the other end they will be connected to the slot pin in the breadboard closest to the corresponding pico pin.



4. Milestones for Each Lab Sessions

The milestones of the second phase can be made in parallel by the different elements of the group or made in a different order as the suggested here. The order given is the one most synchronized with the contents given in the theory lectures, but if the group wants to progress faster, the relevant information will be given per request. Again, the objectives are too challenging to complete during session time (90 min) and equivalent autonomous work will be required.

Session 4 (Mar 28th – Apr. 1st) – Implementation of the I2C Network and Communications

Place all luminaires in the same box (e.g. choose the "best" box of your group). Test the performance of the local controllers in the shared environment (non-cooperative decentralized control). Can they achieve the desired performance?

Assemble the I2C Bus network. Connect all luminaires in via I2C Bus. Write and test a program to send and receive simple messages between controller nodes. Note that you should define a way to assign different addresses to each of the controller nodes.

Session 5 (Apr 4th – 8th) – System Wake-up, Calibration and Hub function

Develop network "wake-up" procedures so that all nodes are aware of each-other and boot properly.

Calibrate of the gains of distributed system model. Write a program to be deployed in all controller nodes to make them synchronously turn on and off the luminaires. This code then should be used to calibrate the self- and cross-coupling gains between the several nodes in the distributed system. This procedure should be done every time the system wakes up to prevent errors in calibration due to movement of the elements inside the box.

Implement the Hub function to serve the specified commands. Write and test a program on the microcontroller that route messages between the PC and other controller nodes. Interface this program with the Serial Monitor.

- **Session 6 (Apr 18th – 22nd) – Distributed Controller**

Implement one of the coordinated distributed controllers given in the theory lectures using C++ classes. Define appropriate communication protocol to exchange the required messages between the nodes. Deploy and test your code in the microcontrollers. Show how the controllers react to changes in desk occupation and external disturbances. Consider two cases: identical or different costs in the energy minimization criteria. Compare with the non-coordinated controller tested in the previous session.

5. Guidelines for the presentation

Demonstrate the ability of the PC interface with the network. Did you implement all commands? Demonstrate the ability of system to properly wake up and calibrate the. Demonstrate the ability of the system to properly control the illuminance of each desk in a cooperative way. Demonstrate the ability of the system to compute energy saving solutions both for identical and different costs of energy in the luminaires. Demonstrate ability of the system to react properly to external illumination. Be prepared to show the implemented code an answer questions about the implementation.

– Enjoy the project –