Instituto Superior Técnico

Applications and Computation for the IoT

3rd Lab work – Project: Intelligent public lighting for smart cities[1]

Goal

The goal of this project is the implementation of a modular and scalable system to control public lights, saving energy but preserving adequate levels of comfort and security. The overall system has some degree of fault tolerance and fail safe behavior.

In the laboratory public lights will be replaced by LEDs, otherwise the system to be implemented exhibits realistic behavior. All public lights have their own controllers which are interconnected via an I2C bus.[2]

Description

The purpose of the project is the design and development of a system to control public lamps in an urban area.

Each set of two lights has its own controller which senses the following events or environmental parameters:

- Detects movements in its area
- Measures ambient light

According to the measured conditions the controller adjusts he brightness of the lamp to generate an appropriate level of luminosity.

The cell formed by a controller and a lamp is identified by its coordinates in a (x, y) plane.

To make better use of the resources available at the laboratory:

- each Arduino controller will drive two lamp cells;
- each lamp will have a specific movement detector;
- the two lamps (two contiguous cells controlled by the same Arduino) will share the same light sensor;

[1] V1.2, 24 January 2022. V1.1, 20 January 2022. V1.0, 16 January 2022.
[2] In a real implementation the I2C bus would be replaced by some form of wireless network – WiFi, LoRaWAN, etc.

- each lamp cell will be identified by its own (x, y) coordinates; an Arduino controller will drive cells (x, y) and (x+1, y);
- all these components must be neatly mounted on the corresponding breadboard.

Besides local control the controller of a cell will broadcast to its eight neighbor cells the following events:

- detection of movement
- failure of a light

and receive from them information about similar events.

Failures of components of the system (e. g. the failure of a lamp) must also be collected by a monitor controller – controller (0, 0) – to support the maintenance of the system.

The communications media is an I2C multi-master bus.

Based on this configuration it is possible to implement intelligent management policies to save energy but preserve an appropriate level of comfort and security:

- maintain appropriate levels of light during day (may be with cloudy weather) and night;
- turn lights fully ON if something (a person, a car) is moving in the cell;
- turn lights ON to a median level around the cell in which movement was detected;
- turn lights ahead following the movement of something (possibly not with full intensity).

The functionalities of each controller must be implemented according to a specified interface protocol so that different implementations of a cell are able to interoperate harmoniously.
(The lamp cells implemented by a group of students must be interoperable with lamp cells implemented by any other group.)

Light cell

A light cell simulates a square with a side of 40 m and with a lamp post in the center. A large urban area will be covered by several contiguous cells. Each Arduino controls two light cells.

Each light cell has one lamp implemented with a LED. An Arduino controller will drive a red LED on cell (x, y), and a green LED on cell (x+1, y);

Each cell has a movement detector. Since the Arduino kits do not have movement sensors (such as ultrasound emitters-receivers) detection of movement can be implemented by the available press button or potentiometer: any transition in the state of the button (ON → OFF, or OFF → ON), or a change in the position of the potentiometer, mean something is moving in the area of the cell.

The light sensor is shared between the two cells controlled by the same Arduino. The intensity of the light has the following states:

- LED ON – Local level by night when there is movement in the cell.
- LED ON/2 – Surrounding (comfort) level by night
- LED ON/4 – Safety level by night.
- LED OFF – Bright day light.

Besides self-adjustment to the activity detected in the area, a cell is also able to adjust the intensity of its lamp to the brightness of the environment, to maintain an adequate level of light in its zone. The levels of brightness of the environment to be considered are:

- Dark night – LED ON; from fully ON to ¼ ON, depending on movements in the area, as it was previously indicated.
- Cloudy day, transitions day to night, and night to day – LED partially ON; environment brightness + lamp light must produce an overall light level corresponding to the 1/4, 1/2, and 100% lighting conditions, according to the movements detected.
- Bright day – LED fully OFF.

Ignoring any events external to the cell, the intensity of the light shall vary linearly and inversely to the detected luminosity of the environment. Bright day light corresponds to the direct (vertical) incidence of the fluorescent light of laboratories.

When movement is detected the LED of the cell will keep its state for 10 s after the event until it turns OFF or decreases brightness. The change in brightness may be abrupt or gradual (in this case continuously changing brightness during 1 s.)

Each cell has a self-check circuit which is able to detect a “cast lamp” (in fact an open circuit LED) and signal the situation to cell (0, 0).[3]

A cell is identified by its (x, y) coordinates. These are setup by wire jumpers connected to input ports of the controller:

- x = 0, a2, a1, a0 (4 bits, 3 least significant setup by jumpers)
- y = 0, b2, b1, b0 (4 bits, 3 least significant setup by jumpers)

The cell (0, 0) receives messages signaling a lamp failure and drives a yellow LED indicating the status of the system:

- LED OFF – After power-up if no faults were registered
- LED ON – When a lamp fault is received
- LED ON → OFF – When the number of “I am OK” messages received equals the number of “Fault” messages.

All cells, regardless of position, must have the same functionality. Even the monitor function can be downloaded in every cell, but only activated in cell (0, 0).

[3] The (x, y) cell sends a message directly to (0, 0) through the I2C bus. In a real wireless sensor network this information would be transferred from cell to cell approaching (0, 0).

Data Interchange

Each cell interchanges data with the surrounding cells through an I2C bus in multi-master mode. The I2C frame has an addressing block and a data block.

The data block has the following format:

- Source (8 bits): (x, y) identifier of the cell which generated the frame.
- Event (8 bits): Type of event or action signaled
    - 0 – time (just sends the current time of the cell in the Time field)
    - 1 – movement (inside the cell) detected
    - 2 – movement with a predictable direction detected
    - 254 – failure of the lamp
    - 255 – I am OK; sent after a failure of a lamp is repaired.
- Time (32 bits): Time stamp associated with the event. The time unit is 10 ms.

The ddressing block preceds the data block and records the destination of the message:

- Destination (8 bits) - (x, y) identifier of the cell to which frame is sent.

Since each controller implements two cells, pairs of cells will have the same I2C address. The relation between cell addresses and I2C addresses is:[4]

I2C address (8 bits) = x/2 | y

Examples:

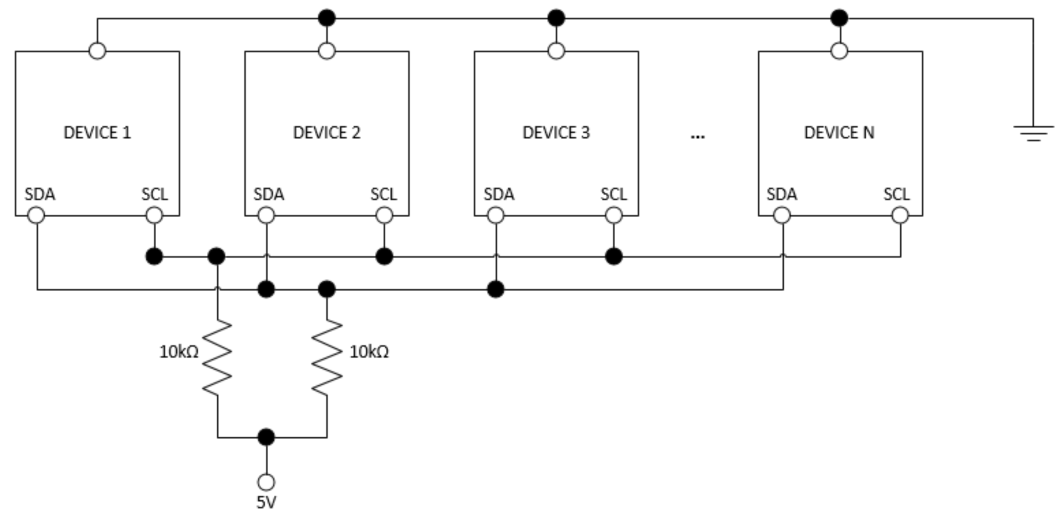
cell address (2, 3) → I2C address (2/2, 3) = 1 | 3 = 1×16 + 3 = 19
cell address (3, 3) → I2C address (3/2, 3) = 1 | 3 = 1×16 + 3 = 19

In each Arduino the communications API must deliver the message to the appropriate cell.

Note that not all messages are interchanged through the I2C bus. Data communications between the two contiguous cells controlled by the same Arduino controller do not flow through the bus. However the application must see the same communications interface (API), whatever the exchange media. In a controller the modules which control each cell mast be independent and communicate exclusively through the communications interface. This interface must check the destination and, either send a message through the I2C bus, or implement some other form of local transfer of data.

The physical connections to implement the I2C bus are depicted in the following diagram.

[4] "/" represents integer division, and "|" concatenation - $abcd_2$ | $cdef_2$ = $abcdefgh_2$.



Light Management – Use Cases

Basic – Static Events

Movement detected inside cell (3, 3):

6	1/4	1/4	1/4	1/4	1/4	1/4	1/4
5	1/4	1/4	1/4	1/4	1/4	1/4	1/4
4	1/4	1/4	1/2	1/2	1/2	1/4	1/4
3	1/4	1/4	1/2	1	1/2	1/4	1/4
2	1/4	1/4	1/2	1/2	1/2	1/4	1/4
1	1/4	1/4	1/4	1/4	1/4	1/4	1/4
0	1/4	1/4	1/4	1/4	1/4	1/4	1/4
	0	1	2	3	4	5	6

1	movement
1/2	comfort
1/4	safety

Non-correlated movements inside cells (3, 3) and (4, 4):

6	1/4	1/4	1/4	1/4	1/4	1/4	1/4
5	1/4	1/4	1/4	1/2	1/2	1/2	1/4
4	1/4	1/4	1/2	1/2	1	1/2	1/4
3	1/4	1/4	1/2	1	1/2	1/2	1/4
2	1/4	1/4	1/2	1/2	1/2	1/4	1/4
1	1/4	1/4	1/4	1/4	1/4	1/4	1/4
0	1/4	1/4	1/4	1/4	1/4	1/4	1/4
	0	1	2	3	4	5	6

1	movement
1/2	comfort
1/4	safety

Detection of Movement

Sequence of movements: inside cell (3, 3) at t1 and inside (4, 4) at t2 (t2 > t1).

Cell (4, 4) evaluates the situation.

Speed ≈ $\Delta l / \Delta t$
$\Delta l \approx \text{sqrt}(40^2 + 40^2) = 57$ m (diagonal movement), or
$\Delta l \approx 40$ m (horizontal or vertical movement)
$\Delta t = t2 - t1$

If s < 20 km/h = 5.6 m/s consider a slow movement → control lights as non-correlated movements.

If s ≥ 20 km/h = 5.6 m/s, turn ON lights ahead in the direction of movement (linear forecast):

(3, 3) → (4, 4) → (4+1, 4+1) = (5, 5) .

Cell (4, 4) sends to (5, 5) a message "Movement with a predictable direction".

For a diagonal movement corresponds t the condition t2 – t1 ≤ 10.2 s:

6	1/4	1/4	1/4	1/4	1/4	1/4	1/4
5	1/4	1/4	1/4	1/2	1/2	1	1/4
4	1/4	1/4	1/4	1/2	1	1/2	1/4
3	1/4	1/4	1/4	1/2	1/2	1/2	1/4
2	1/4	1/4	1/4	1/4	1/4	1/4	1/4
1	1/4	1/4	1/4	1/4	1/4	1/4	1/4
0	1/4	1/4	1/4	1/4	1/4	1/4	1/4
	0	1	2	3	4	5	6

1	movement
1/2	comfort
1/4	safety

Clock Synchronization

To predict the direction of movement it is necessary to estimate the speed associated with a sequence of events and, therefore, it is required a common time reference. Since the Arduino controllers are fully independent and may be powered up at different moments each cell needs to execute an initial procedure to "align" the local clock, obtained from the millis() function, with the other cells.

The first 5 s after reset are reserved for this and other initialization procedures.[5]

Synchronization sequence:

1. Each cell (x, y) sends its clock to up to 4 neighbors – North (x, y+1), South (x, y-1), East (x+1, y), and West (x-1, y).
2. Each cell receives the clocks from its 4 neighbors, or from the existing neighbors.
3. Evaluate the clock differences.

   For each neighbor

   $\Delta i = ti - tK$,

   Δi clock difference, ti clock of the cell, tK clock of the neighbor, ND network delay.

4. Average the differences and adjust the clock.

   $delta = \sum \Delta i / n$,

   n number of effective neighbors,

   new_time = millis() – delta

Example

A configuration with 4 cells, each with its independent clock. With this layout each cell has only two neighbors at cardinal points. When the sequence starts the clocks of the cells are misaligned 100 U (time units).

Each cell receives the message from its neighbor 10 U after it was sent (the Network Delay):

Cell (1,0) receives a message time stamped 1000 from cell (0, 0) at local time 1100 + 10 = 1110 U → Δi = 1110 – 1000 = 110 U.

4 cells	(0,1)	(1,1)											
	(0,0)	(1,0)											
	ND (network delay)			10									
		Receives message											
Sends message		Cell	@ t	Δ	Cell	@ t	Δ	Cell	@ t	Δ	Cell	@ t	Δ
Cell	@ t	(0,0)			(1,0)			(0,1)			(1,1)		
(0,0)	1.000				1.000	1.110	110	1.000	1.210	210			
(1,0)	1.100	1.100	1.010	-90							1.100	1.310	210
(0,1)	1.200	1.200	1.010	-190							1.200	1.310	110
(1,1)	1.300				1.300	1.110	-190	1.300	1.210	-90			
delta				-140			-40			60			160
new_time				1.140			1.140			1.140			1.140

[5] Other initializations requiring a longer period (possibly the calibration of the light sensor) shall not interfere with this synchronization process.

Recommendations

In order to fulfill your work with security and not damaging the hardware involved, remember to carry out the recommendations below. As you are working fill the boxes to be certain that you fulfill all security measures.

Always work with the circuit disconnect from the source.	
Call the professor or responsible for the laboratory, before you connect the circuit to the source.	
Make sure the circuit is well connected (resistors, capacitors, etc.) to prevent a short circuit, or damage the hardware.	

Plan and Deliverables

1 week = 3 h class. Each section corresponds to a 1.5 h period.

Week	Activity
1.1	Project presentation.
1.2	Requirement analysis and specification of the data interchange formats of I2C messages. Project design and development.
2	Specification of the data interchange format. Project design and development.
3.1	Demonstration of lamp control in a cell and interoperation of a set of four cells. (Presence of all the members of the group required.) Delivery of the project report (printed report, and e-mail), including 1. design of the circuits of the controller and lamps, 2. overall architecture of the software in the controller and lamps, 3. safety and fault-tolerance measures adopted, 4. “intelligent” functions implemented, 5. programs (properly commented) implementing the system. (The report includes interoperable functionalities to be demonstrated in the next phase.)
3.2	Demonstration of the integrated system controlling public lights by sets of student groups to be defined. (Presence of all the members of the group required.) Each set will have 3 – 4 groups of students, each group handling four lamp cells. After the class each set of groups delivers (e-mail) a brief evaluation report of the integration test, to complete, correct, or confirm the reports delivered previously delivered.

Week	Activity
4	Oral evaluation of the groups. Covers all laboratory works and the project. (Presence of all the members of the group required.)