Instituto Superior Técnico

Master's Degree in Information Systems and Computer Engineering

Applications and Computation for the IoT

4th Lab work – Project: Intelligent control of traffic flow[1]

Goal

The goal of this project is the implementation of a modular, automatic and distributed control of traffic lights at intersections. The system is able to adapt the control of the traffic lights to the variations in traffic, to let traffic flow faster on the busiest roads.

The overall system has some degree of fault tolerance and fail safe behavior.

In the laboratory traffic lights will be replaced by LEDs, and traffic sensors by buttons or switches. Otherwise the system to be implemented exhibits realistic behavior. All the peripherals on an intersection are connected to a controller. Intersections exchange information through an I2C bus.

Description

The purpose of the project is the design and development of a traffic lights system to control traffic at intersections.

Each intersection has the layout depicted in Fig. 1 forming a simples cross. Each access accepts vehicles in both directions and there are no restrictions to the movement of vehicles in the intersection, beyond the basic rules of right-hand traffic. Accesses are identified by their orientation:

- N – North (upper access);
- S – South (lower access);
- E – East (right access);
- W – West (left access).

Each intersection has two traffic lights at orthogonal accesses. In the figure, TL_S controls accesses of vehicles approaching the intersection from South, and TL_W controls vehicles approaching from West. Traffic flow in these accesses is monitored by inductive loop detectors

[1] V2.0, 11 November 2019. V1.0, 7 November 2019.

(ILD, or LD) embedded in the road pavement and simulated by some form of switch (any device able to toggle a signal from low-high-low – a press button, a potentiometer, a jumper).

To simplify the hardware assembly we will not implement the traffic lights and sensors in the remaining two accesses, and will evaluate traffic flow based on the information gathered in the in the two accesses emulated.

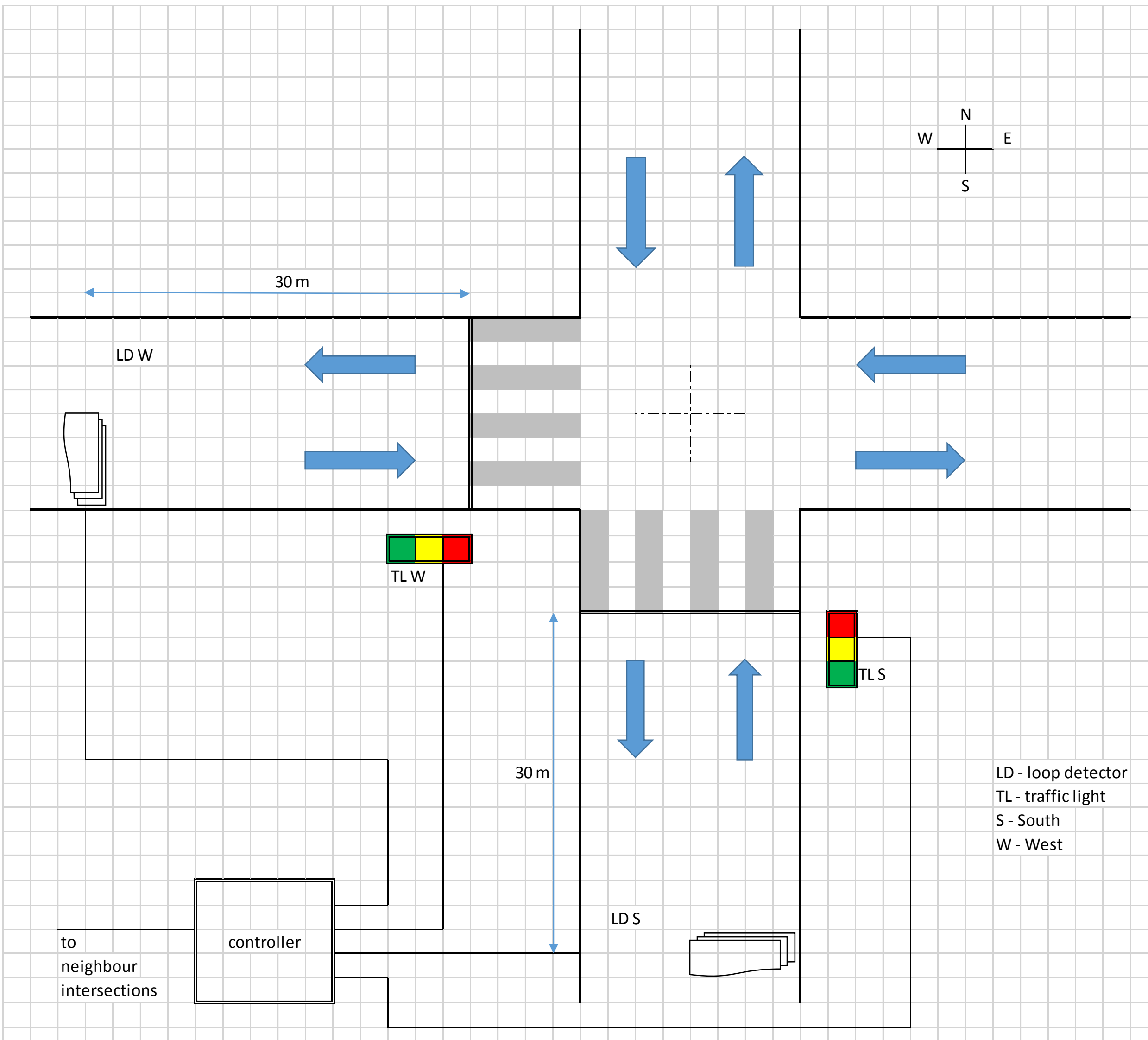

Fig. 1 – Intersection.

With only two accesses equipped with sensors (LDs) ans actuators (traffic lights) intersections have four possible orientations, represented in Fig. 2.

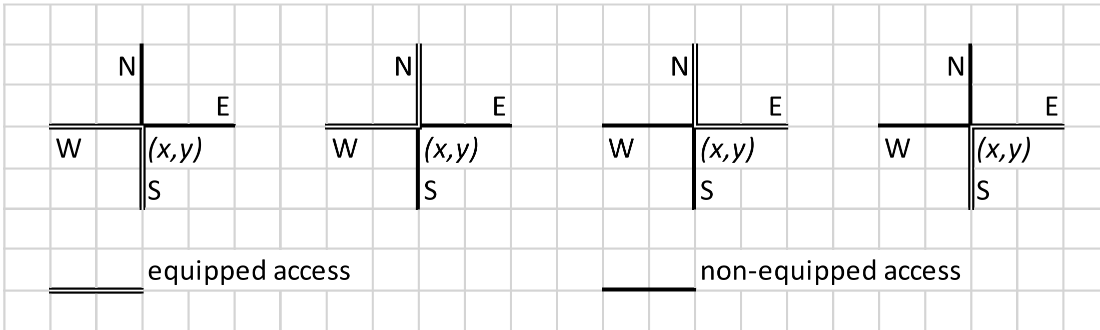

Fig. 2 – Orientations of intersections.

Traffic Light

Every traffic light must have 3 LEDs – green, yellow, and red. The intersection controller must be able to detect the failure of any red LED.

Loop Detector

Inductive loop detectors (LD) embedded in the pavement of the road are a common sensor used to detect the passage of the vehicles over them. The detection mechanism is based on the variation of the frequency of a resonant circuit when the metallic mass of a vehicle passes over the coil embedded in the pavement and changes its equivalent inductance.


Fig. 3 – Inductive loop detectors for cars and bikes.[2]

Considering that an LD generates a voltage HIGH when a car is passing over and a voltage LOW otherwise, its function can be simulated by any device which is able to toggle a signal from low-high-low – a press button, a potentiometer, a jumper, etc. LDs can be used to count

[2] Induction Loop. https://en.wikipedia.org/wiki/Induction_loop . Acessed Nov. 2019.

vehicles, and sets of LDs can measure speed and classify the type of vehicle by evaluating its size.

Controller:

The controller must be designed as an Arduino UNO with the I/O devices attached.

- 2 × 3 LEDs, the traffic lights;
- 2 LD simulators (press buttons, potentiometers, etc,);
- the I2C bus interface to communicate with the controllers of neighbor intersections.

The identification of the intersection is specified by its (x, y) coordinates in space. These are setup by wire jumpers connected to input ports of the controller:

- x = a3, a2, a1, a0 (4 bits)
- y = b3, b2, b1, b0 (4 bits)

Requirements

1) **Initial state** of the system (at an intersection) – all traffic lights blinking yellow with a 2 seconds period (ON + OFF cycle time) during 6 seconds. After this the system enters normal operation.
2) When a malfunction is detected (red LED failure, software fault) the system will go to a **standby state** stopping all functions except blinking the yellow LEDs with a 2 seconds period (Fig. 3).
3) **Normal operation** has three modes of functionality:
   a) **Mode 0 –** Traffic lights switch between red and green with a fixed duty cycle, without any consideration of traffic density.

      Period = 20 seconds, with 50% duty cycle.

   b) **Mode 1** - Traffic lights switch between red and green with a duty cycle adjusted according to the relative levels of traffic in each access.

      Period = 20 seconds

      Duty_cycle_S = cars_S / (cars_S + cars_W)
      Duty_cycle_W = cars_W / (cars_S + cars_W)
      (case of an intersection equipped in accesses S and W.)

      bounded between

      $$5/20 = 25\% \leq \text{Duty cycle} \leq 15/20 = 75\%$$

      to avoid blocking one road due to traffic on the other.

c) **Mode 2** - Traffic lights switch between red and green with a duty cycle adjusted according to the relative levels of traffic in each access and the information gathered from the neighbor intersections.

Besides varying the duty cycle of traffic lights according to the traffic densities – like in mode 1 – in mode 2 the controller tries to adjust the phase of lights to enable along a street with a high level of traffic the movement of vehicles without "disturbing" red lights.

Modes 0 and 1 are purely local modes which evaluate only information from local LDs. (Mode 0 does not even read LDs.) Mode 2 is a distributed control mode in which all the controllers in an area cooperate to improve traffic flow.

4) All communications between intersections must be performed using I2C protocol.

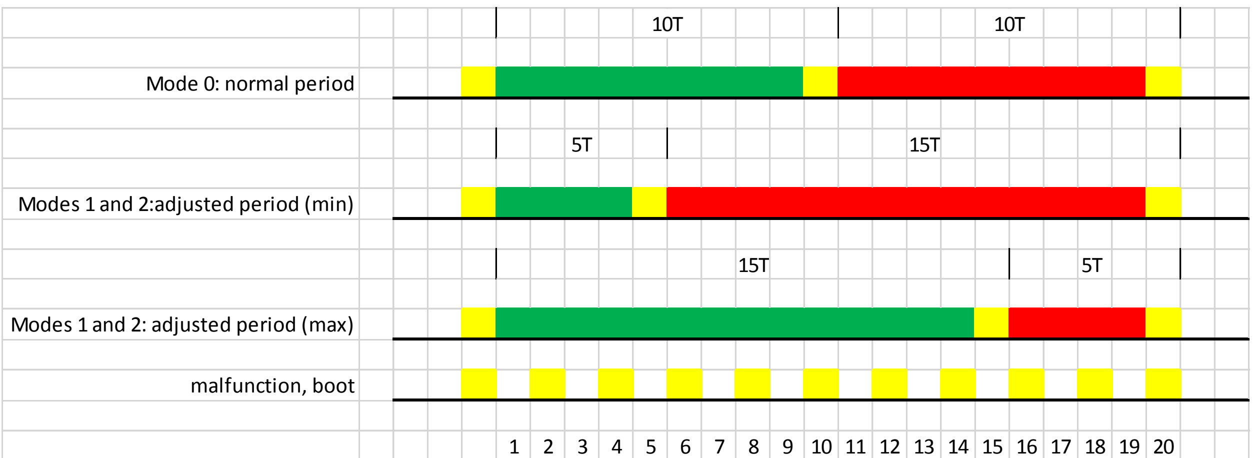

Fig. 4 – Traffic lights operation.

Data Interchange

Each intersection interchanges data with the surrounding cells through an I2C bus in multi-master mode. In every control period the controller of an intersection sends messages to its neighbors informing its state. The I2C frame has an addressing block and a data block.

The addressing block records the destination of the message:

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

The data block has the following format:

- Source (byte): (x, y) identifier of the intersection which generates the frame.
- Event (byte): Type of event or action signaled. In all cases "red to green" corresponds to "yellow-after-red to green" in Fig. 4;
  - 0 – R2G_N (red to green in the North access);
  - 1 – R2G_S (red to green in the South access);
  - 2 – R2G_E (red to green in the East access);

- 3 – R2G W (red to green in the West access);
  - n – (n > 3, reserved).
- Cars (structure with four bytes)
  - Cars_N (byte): cars counted at North access during the last period;
  - Cars_S (byte): cars counted at South access during the last period;
  - Cars_E (byte): cars counted at East access during the last period;
  - Cars_W (byte): cars counted at West access during the last period;
- Time (32 bits integer): Time stamp associated with the event. The time unit is 100 ms.

The physical connections to implement the I2C bus are depicted in Fig. 5.

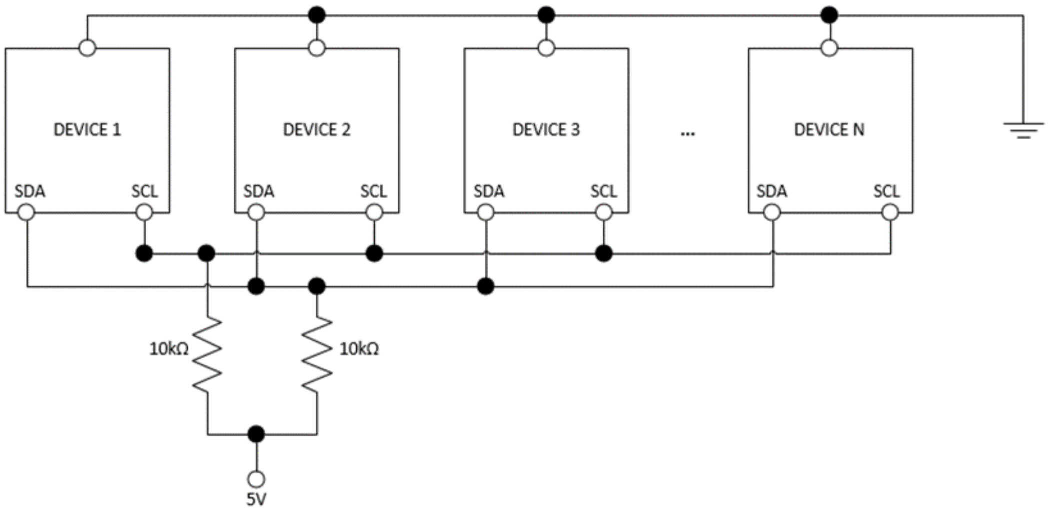

Fig. 5 – I2C Multi-master diagram.

Distributed Control

Most of the complex urban traffic control systems nowadays have a centralized controller which oversees the operation of traffic lights. In the project we will prototype a distributed control system. To simplify the implementation we will restrict the layout of the urban area in which the system will be "installed" and develop a simple control algorithm.

We will consider a regular urban plan with similar blocks on an orthogonal net of streets (Fig. 6). All blocks have the same size and the distances between neighbor intersections are all the same.

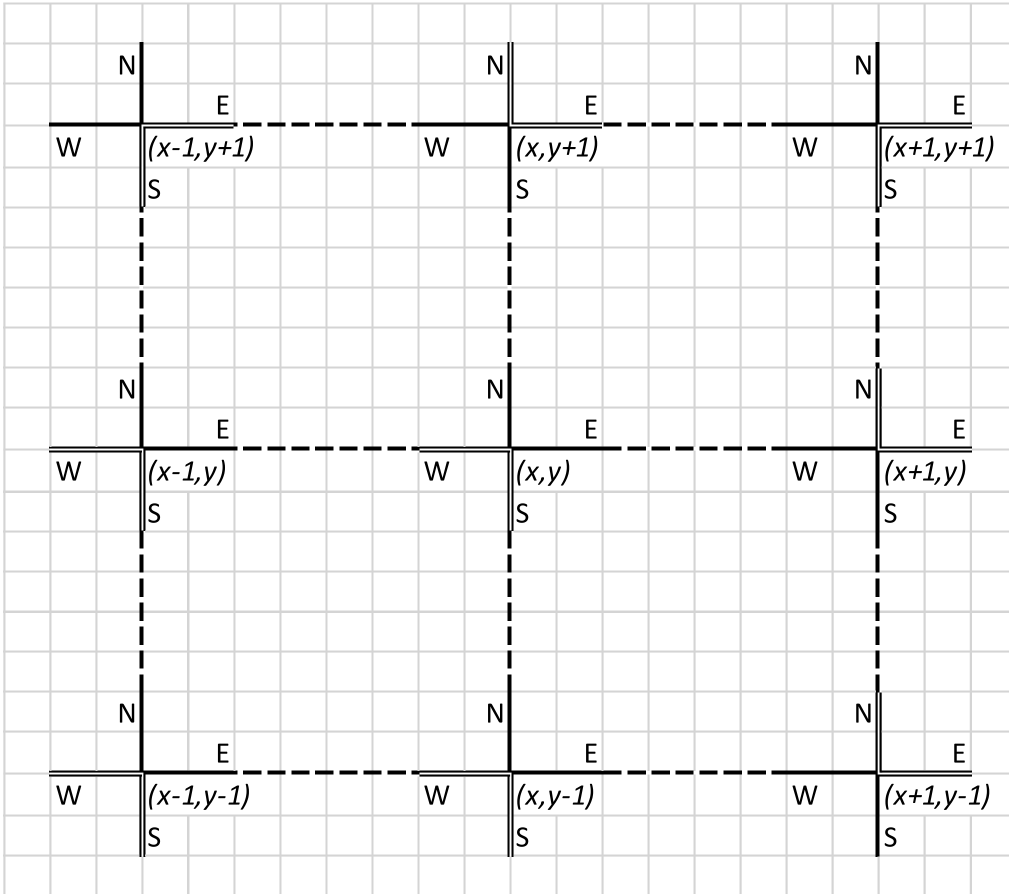

Fig. 6 – Regular urban plan with orthogonal streets. (Intersections may be equipped in any orientation.)

In each moment the distributed algorithm tries to maximize traffic flow along streets with the highest level of traffic. This is achieved with:

- Control of duty cycle – manages the duty cycle of the traffic lights in each intersection, based on the local information, and
- Control of phase – adjust the phases of traffic lights at adjacent intersections to create a green wave letting cars flow which less disturbance; if cars take 12 seconds to go from an intersection to the following then the green wave will look like Fig. 7.

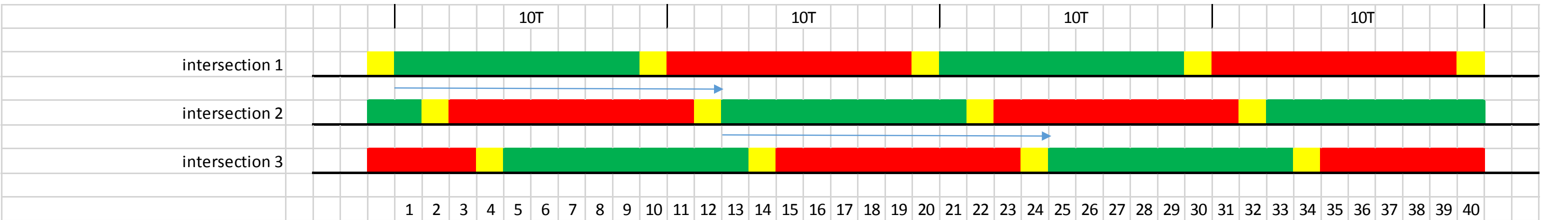

Fig. 7 – Example of a green wave. Cars move from intersection 1 to 3.

Control of duty cycle is already supported in Mode 1 of operation.

Control of Phase

After receiving a message from a neighbor the controller of the intersection checks if the direction of maximum traffic coming from the neighbor is aligned with its own maximum traffic. If so it will try to adjust the phase to form a green wave. If there is no alignment it keeps unchanged its pattern of control of its traffic lights.

Examples (refer to Fig. 6):

(x-1, y) →(x, y)

1. (x, y) controller receives a message from (x-1, y).
2. Maximum flow from West in (x-1, y) = Maximum flow from West in (x, y)?
3. YES: (x, y) will begin to adjust the phase of its control period to form a green wave.

(x, y) →(x-1, y)

1. (x-1, y) controller receives a message from (x, y).
2. Maximum flow from West in (x, y) = Maximum flow from West in (x-1, y)?
3. IGNORE: traffic in the opposite direction (from East to West, and not "from West").

The ideal adjustment delays the transition from red to green (transition from yellow-after-red to green) at the next intersection by the time needed for a car to reach the intersection, however this cannot be done abruptly to preserve (or only slightly change) the duty cycle of the traffic lights. Therefore the system must implement the adjustment of the phase in three steps (Fig. 8).

We will assume that the system knows the typical time for a car to move between two consecutive intersections. (6T in the figure).

- t = 0
  - Controller 1 sends a message to controller 2.
  - Controller 2 evaluates the phase difference = 6 but it has already started the period.
  - Controller 2 slightly increases the period +1T green + 1T red. Next green delayed 2T.
- t = 20
  - Controller 1 sends a message to controller 2. (The alignment conditions are unchanged.)
  - Controller 2 evaluates the phase difference = 6 – 2 = 4 but it has already started the period.
  - Controller 2 slightly increases the period +1T green + 1T red. Next green now delayed 4T.
  -
- t = 40
  - Controller 1 sends a message to controller 2. (The alignment conditions are unchanged.)
  - Controller 2 evaluates the phase difference = 6 – 4 = 2 but it has already started the period.
  - Controller 2 slightly increases the period +1T green + 1T red. Next green now delayed 6T → green wave.

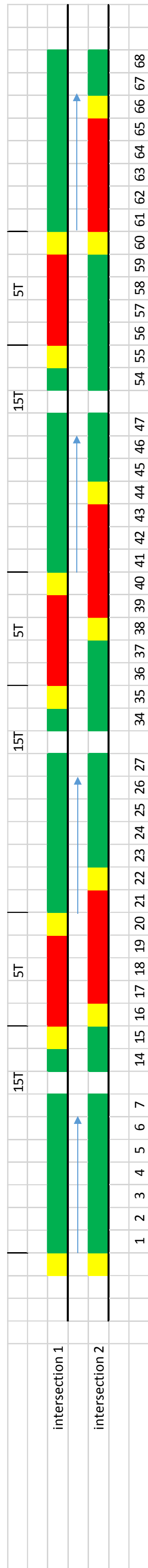

Fig. 8 – Phase adjustment to create a green wave.

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

Week	Activity
1	Project presentation.
2	Requirements analysis and specification of the data interchange formats of I2C messages. Project design and development.
3	Specification of the data interchange format. Project design and development.
4	Demonstration of two contiguous intersections by each group. Delivery of the project report (printed report, and e-mail), including a. design of the circuits of the controller, traffic lights, and sensors, b. overall architecture of the software in the controller, c. safety and fault-tolerance measures adopted, d. “intelligent” functions implemented, e. programs (properly commented) implementing the controller and traffic lights. (The report includes interoperable functionalities to be demonstrated in week 5.)
5	Demonstration of the integrated system controlling a city area by sets of student groups to be defined. Each set will have 3 – 4 groups of students, each group handling two intersections. Each group delivers (printed report, and e-mail) a brief evaluation report of the integration test, to complete, correct, or confirm the report delivered in week 4.
6	Oral evaluation of the groups. Covers all laboratory works and the project.