Project
Real-Time Computer Systems [ELEC-H-410]
Design a real-time distributed alarm

2017–2018

1 Purpose

On the basis of the previous labs you will now create a distributed alarm system working on the CAN network.

2 Useful documents are stored on the network share:

```
\\labo\ELEC-H-410\Useful Documents\
  - dsPIC30F-33F Programmer's Reference.pdf
  - dsPIC33 Data Sheet.pdf section 21
  - Introduction to MPLAB.pdf
  - Explorer 16 User Guide 51589a.pdf
  - MPLAB C30 C Compiler User's guide.pdf
  - uCOSII_RefMan.pdf
  - Enhanced Controller Area Network.pdf
  - Introduction to language C for microcontrollers.pdf
  - Table of ASCII codes.
  - Troubleshooting µC/OS-II
```

3 Specifications

In this distributed alarm system, all nodes are equal (i.e. there is no master node), can work autonomously but also exchange information through CAN. The application of the node should run under µC/OS-II.

Each node of the alarm system will have to carry the following functionalities out:

- Monitor the presence of intruders. The sensor will be a push button connected to a digital input of the microcontroller.
- Arm/disarm the system with a secret code typed on the keyboard. For the moment, you can use the value B169. Functions to change this code are optional.
- If an intruder is detected while the system is armed, leave the possibility to disarm the system using the same code during the first 30s.
- If the system has not been disarmed within 30s, the alarm should start. You will get a buzzer or a flashing light that you can connect to an output port.

In a distributed system, each node must cooperate with the other members of the network:

- A node signals its presence when it appears on the network and repeat this message periodically at least every 5 seconds. This "heartbeat" enables to detect the disappearance of the node (failure or sabotage).
- If a node detects an intrusion, it must signal it so that all other nodes pass in the "alert" state, even if didn't detect anybody.
- (Dis)arming a node should be propagated to all other nodes; there is no obligation to disarm at the node which has detected the intrusion.
- Think about any way to attack your alarm and try to reduce possibilities of success in doing that.

To facilitate the management of the communication, we suggest that you use the following protocol:

- `Message_ID` will be fixed during the first brainstorm and is linked to the priority of the messages on the network. Nodes must specify their `Node_ID` in the data, one byte, value 0 to 255.
- The alarm system is limited to 10 nodes whose `Node_ID` is unknown a priori.
- Then a node boots, it has to find a free identifier, then signal its presence by broadcasting it regularly.

Message type	**Message_ID**	**Data**
Heartbeat	0	node ID (one byte)
Intrusion detected	1	node ID
Node has been armed	4	node ID
Node has been disarmed	2	node ID
Alarm started	8	node ID
New password	24	node ID + password (4 characters)

Table 1: Table of messages

4 Operating mode

The project will be done in group of 3 or 4.

4.1 Timetable

Five sessions are dedicated to the project:

- During the *first project session*, you have to brainstorm about the specifications of a distributed alarm. Don't hesitate to use timing diagrams, block diagrams, state machines or other operating/functional schemes to share your ideas. You must **NOT** code anything during this session.
- During the *next three sessions*, you will use your knowledge and the result of your brainstorming to implement your node of the distributed alarm system.
- Your project will be evaluated during the *last session*.

4.2 Evaluation

The project will we evaluated on the basis of a short presentation of your work and your answers to questions.
F. Quitin and S. Mezhoud will spend several minutes in each group during the last session to check the effective functioning of your alarm, to assess the quality of your code and to ask you a few questions.

Good luck!