An Alarm System

Group 6
Yunyi Huang PN 881116-7306
Gongyi Zhang PN 870628-T410
Thiago Costa Porto PN 851008-T972
Sven Larsson PN 511216-6938

Objectives

according to Project Plan

- It shall be possible to switch the alarm on, or off, preferably using the touch screen. The alarm shall make some kind of nose if a sensor indicates an alarm condition.
- When the alarm is activated by a door opening, it shall be delayed a configurable short time before it gives a signal. It shall be possible to switch off an activated alarm within this delay. When switching off the alarm the system shall require a code to be entered before actually switching off.
- There is no need for any delay except for the door sensors. So when the other sensors indicate an error the alarm shall go on, giving a signal by making a noise.
- There shall be at least one door sensor.
- When an alarm is activated by a sensor the touch screen shall display which circuit that gave the alarm.

Objectives

according to Project Plan

- Minimum functionality would include:
    - Using the touch screen to switch on and off alarm.
    - Using the touch screen to give a secret code to deactivate the alarm.
    - Using the touch screen to read and clear alarm status.
    - Using the touch screen to configure the waiting time.
    - Using the touch screen to configure the secret code.
    - Door open/close sensor .
    - Some kind of noise making device.

Objectives

according to Project Plan

Additional functionality, if time permitted:

- Additional door sensor.
- A sensor to detect moving objects.
- Temperature sensor.
- Other type of sensors (e.g. light, humidity).
- A camera.
- A phone.
- Different alarm profiles.
- Remote control of the alarm.

Results

according to minimum functionality

- ✓ Using the touch screen to switch on and off alarm.
- ✓ Using the touch screen to give a secret code to deactivate the alarm.
- ✓ Using the touch screen to read and clear alarm status.
- ✓ Using the touch screen to configure the waiting time.
- ✓ Using the touch screen to configure the secret code.
- ✓ Door open/close sensor . Two, different types.
- ~~Some kind of noise making device.~~ Expensive! Using LED instead.

Results

according to additional functionality

- ✓ Additional door sensor.
- A sensor to detect moving objects.
- ✓ Temperature sensor.
- Other type of sensors (e.g. light, humidity).
- A camera.
- A phone.
- Different alarm profiles.
- Remote control of the alarm.

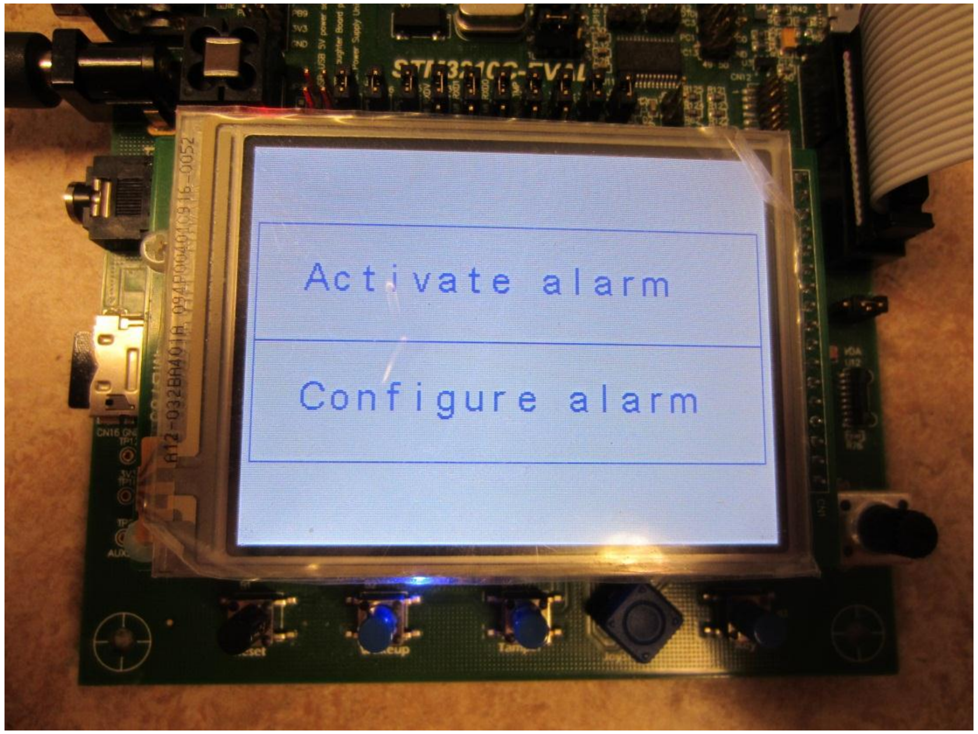
Notes about configuration

- Door sensor 1 (white): connect between GPIOA, pin5 and GPIOA, pin 10.
- Door sensor 2 (grey): connect between GPIOA, pin 6 and GPIOA, pin 11.
- Alarm “signal”: connect between GPIOA, pin 12 and ground on CN8, pin 1.
- Configuration is yet unclear on “Temperature Sensor”. Using the potentiometer (maybe PA1)

Notes about usage

page 1 (4)

- Make the connections and power on.
- On first menu, select Activate alarm or Configure alarm.



Notes about usage

page 2 (4)

Activated alarm:

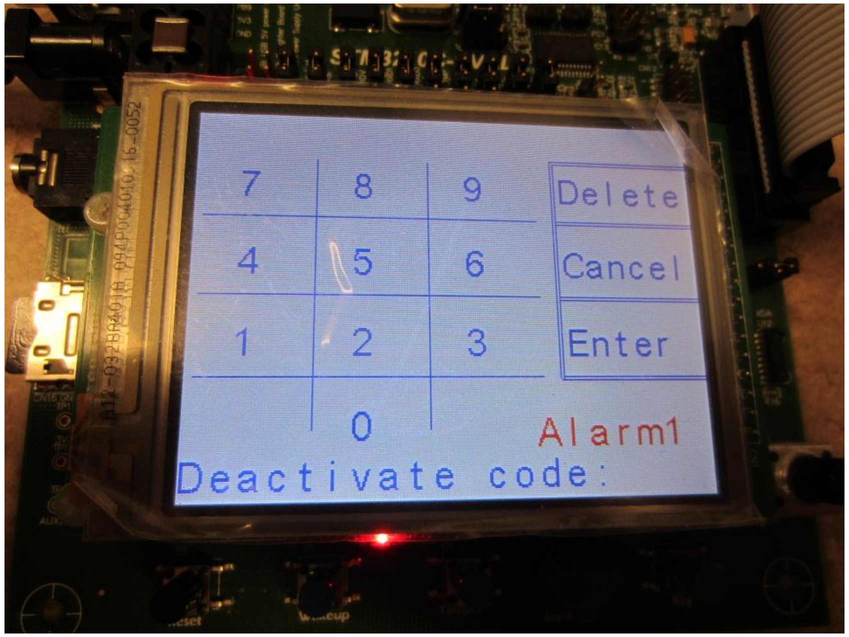
- Alarm sensors activated is indicated by yellow blinking LED.
- When an alarm sensor signals an error it is first indicated by a blinking red LED.
- After the time set in Configuration - Signal delay, the LED will be steady red and a steady signal on the Alarm “signal”.

Notes about usage

page 3 (4)

- The alarm can be reset by a secret code, the deactivation code.

Here is an alarm on sensor 1 (with red LED)



Notes about usage

page 4 (4)

- The configuration alternative first requires a code, to authorize the users right to change the configuration.
- Then there are questions about Secret code and Signal delay.

Status indicated by the LED's:

- Blinking blue: Not activated, someone at home.
- Blinking yellow: Activated alarm. No alarm yet.
- Blinking red: Alarm from sensor. Still within grace period.
- Steady red: Alarm from sensor.

Experiences

- It was a rather good project as it was possible to add functionality depending on time. Although, difficult to find suitable and not expensive add on items (e.g. camera, IR-sensor).
- It was difficult to estimate required time because of new environment.
- We became more familiar with design and coding in C.
- It was a good opportunity to become familiar with boards, embedded systems programming and going through pages and pages of manuals and code

Main Difficulties

- The board was required if wanting to test some code. It took quite a while making it possible to use the debugger without the board.
- We had some problems when trying to use the GPIO pins. Most of them were already occupied and it was difficult to find information on which was free. The configuration of the GPIO's were also a bit different compared to previous assignments.

Main Difficulties

- It was in general rather difficult to find detailed information regarding the board and how to access its functions.
- Process of identifying what and how components need to be setup is very time consuming
- External problems might happen! (Plan B? C?)
- How to save password (continued).

Main Difficulties

How to save password

- If one wants to access to the board, we need a safety authentication system, but how?

1. Memory
2. EEPROM
3. SD card

Main Difficulties

How to save password

Memory

- STM3210C has RAM for 64KB
- Advantage: easy to implement, size is a little bigger than EEPROM.
- Disadvantage: No data can be saved after power is off

Main Difficulties

How to save password

EEPROM

- STM3210C has one EEPROM on the board
- Interface: I2C
- Size: 64Kbit
- Advantage: Data can saved even if the power is off. I2C is easier than SD card interface
- Disadvantage: Memory size is not enough for saving large amount of data

Main Difficulties

How to save password

SD card

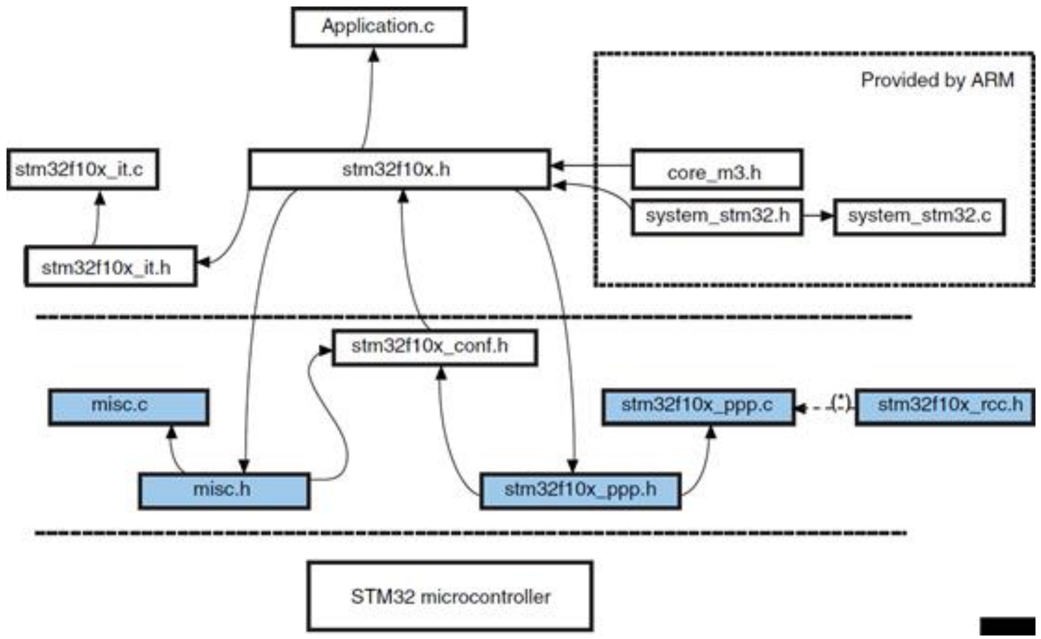
- STM3210C has MicroSD card interface based on SPI.
- Advantage: Larger volume SD card can be supported(around 2G), data can be saved permanently.
- Disadvantage: Operation is too complex, need send lots of commands before it really saves data.

Main Difficulties

How to save password

First try SD card

- By using peripheral library, we can directly use the sample code provided by STM. The relation between different files is showed below:



Main Difficulties

How to save password

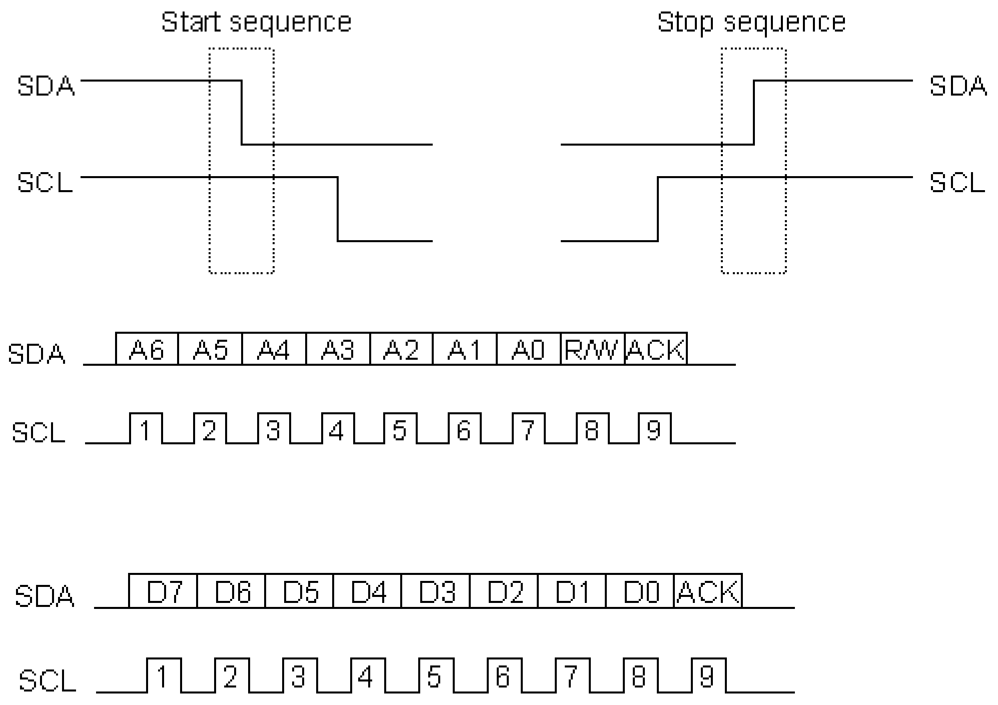
Problem with SD

- Code version for SPI interface is only for 4.1 plus(our environment is only based on version 3.1)
- SPI Commands for SD is too complex
- So we give up!

Main Difficulties

How to save password

EEPROM-I2C



http://www.robot-electronics.co.uk/acatalog/I2C_Tutorial.htm

Main Difficulties

How to save password

EEPROM

- Address for EEPROM in our board: 0xA0
- Our writing address for data starts from 0x49
- Two main functions are used in our implementation:
- void I2C_EE_BufferRead(uint8_t* pBuffer, uint16_t ReadAddr, uint16_t NumByteToRead)
- void I2C_EE_BufferWrite(uint8_t* pBuffer, uint16_t WriteAddr, uint16_t NumByteToWrite)

Amount of Time Needed

We estimate that we have in total spent about: 400+ hours on this project.

Lessons Learnt

- More knowledge on programming the board and RTOS.
- It is essential to make a description of the project and the goals.
- To meet and follow up on a regular basis.
- External tools are needed for similar projects.
- To search for information about required drivers and include files.
- **A lot of things are possible to do with the board – but it takes time!**