# The Pop Button

Kentin and Nil

# Introduction

Do you know Sonia?
Sonia is a young mum, her little baby is sick and she needs to buy some medication. So she, as everyone in 19th century, grabs a piece of paper and notes down the medicine along other things as the eggs she needs for Patrick's birthday's cake...
The day after, at the grocery she realizes that she left the list on the fridge and forgot about the medication, once she came back at home the baby was dying and the neighbour had called the police... She is now serving 5 years at Guantanamo for medical negligence.

Did you ever experienced something similar ???

Do you want to go in Guantanamo ???

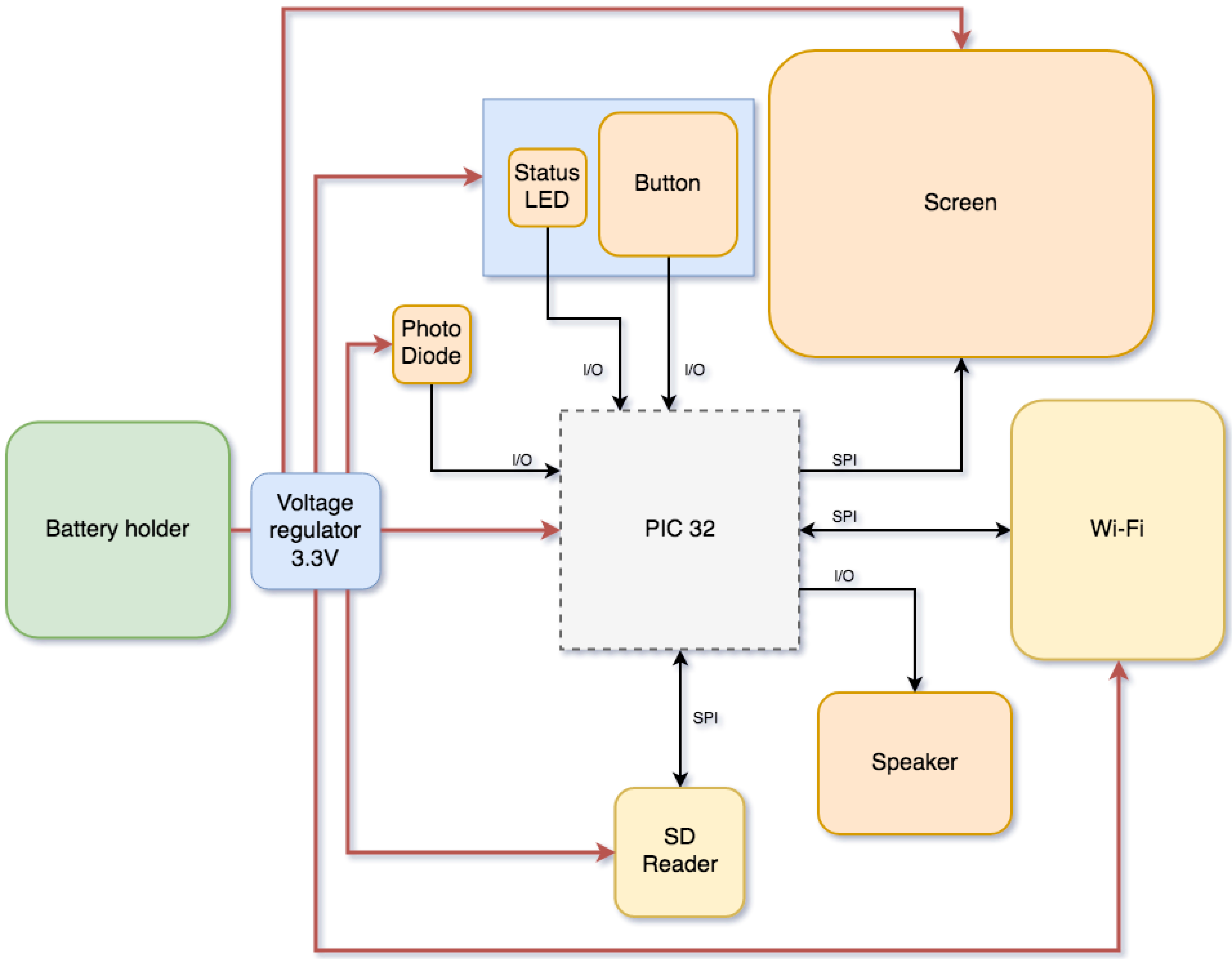
Don't worry we have the solution :

## The Pop Button !

The POP Button is a little device that fits in your cosy place and allows you to keep track of whatever item you need on the cloud in order for you manage it the most efficient way.

If Sonia would have had a POP Button, once she realized she forgot the list on the fridge, she would have been able to connect with her smartphone to our server and find back all the items she entered thanks to her POP friend.

# Block Diagram



# Manual

## First Use / Configuration

Before using the POP Button you have to configure the SD Card, connect the SD Card to a computer and use our program to format it. To use the product for the first time open the rear panel to reveal the battery and SD Card slot. First, put the SD Card in and then put the AA-type batteries. Finally, make sure the switch is in the On position and close the back panel.
Now the screen should light up and you can start to use your product normally. If it doesn't, refer to the troubleshooting section.

## Normal Use

Usually, the screen will be turned off and the device will be in sleep mode. The screen should light up when you approach your hand to the device, if it doesn't, press the knob to turn on the screen.
Once the screen is lit up, you can select the product you want by turning the knob to change the selected item.
Once you're done, the screen will show you a confirmation and you will hear a sound to confirm the order, also the knob will change it's color accordingly.

## Settings

Most advanced settings are edited via the SD Card. But some of them can be edited through the machine. To enter settings, long-press the knob. Once you are in the menu, select the desired option and press the button. Now you can adjust the value and push again to validate.

## Knob Color Codes

| | | |
|---|---|---|
| Off | Sleep | When the button is off it means that the device is in sleep mode |
| Blue | Ready | The device just turned on |
| Orange | Working | The order has been recorded but the device is still sending it to the server |
| Green | Done | The order was processed successfully |
| Red | Error | The device encountered an error |
| Blinking Red | Low Battery | The device is running out of battery. The screen won't turn on until you replace the batteries. |

## Troubleshooting

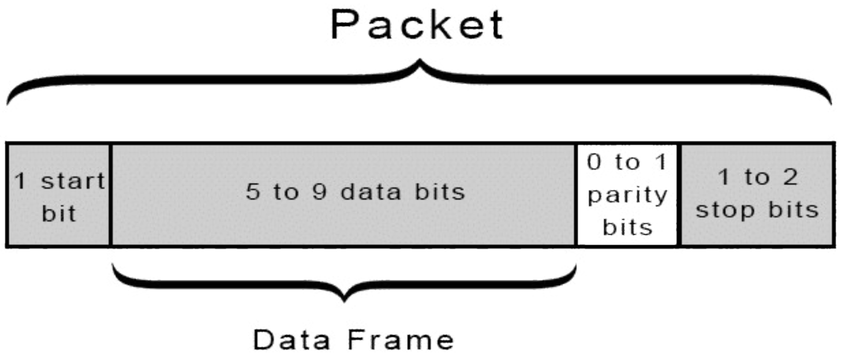
Almost all of the errors can be resolved by following the on-screen instructions. However, here are some other problems that might arise:

| Symptom | Solution |
|---|---|
| The device won't turn on | Check that the switch on the back is turned on. |

# Protocols

## UART

UART stands for "Universal Asynchronous Receiver Transmitter" it is a little circuit in a micro-controller which transmits and receives serial data. As it doesn't use clock it needs to work on a fixed frequency known as the baud rate, the baud rate is explained as bits per second (bps) and both peripherals needs to work on the same one in order to communicate. UART transmits data in packets which consist of 1 start bit, 5 to 9 data bits, an optional parity bit, and 1 or 2 stop bits.




## SPI

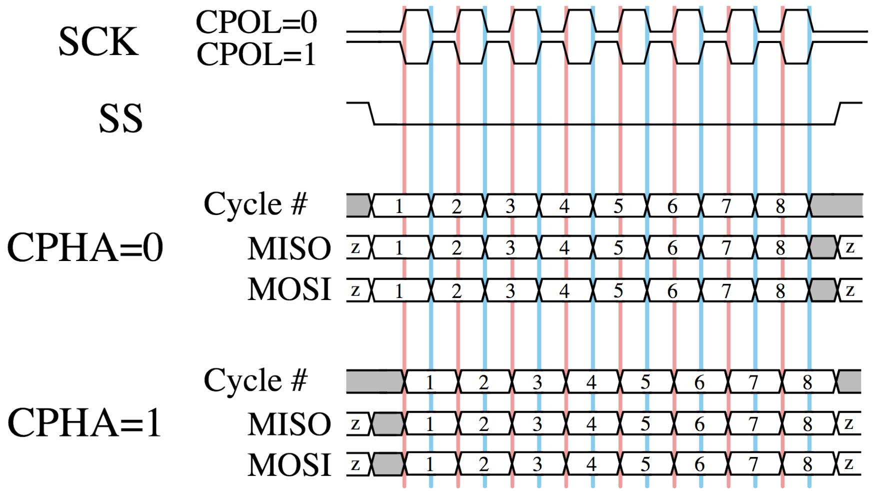
SPI stands for Serial Peripheral Interface it is a synchronous and full-duplex communication protocol, which means that it uses a clock and it can receive and send at the same time This protocol uses 3 lines to work and then one pin more for each slave to in order to select it:

- SCLK ⇒ Serial Clock
  (SCK, SCL)
- MOSI ⇒ Master Output, Slave Input
  (SDI, DI, SI)
- MISO ⇒ Master Input, Slave Output
  (SDO, SDA, DO, SO)
- SS ⇒ Slave Select
  (nCS, CS, nSS, STE, CSN)

So first the master has to select a slave, to do so it lowers the appropiate 'SS' line. To send the data they must have agreed in the settings for Clock Polarity and Clock Phase. There are 4 modes available:

| SPI mode | CPOL | CPHA |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 2 | 1 | 0 |
| 3 | 1 | 1 |

The master then turns on the clock. To send the data, the master uses the 'MOSI' line and the slave uses the 'MISO'. They send one bit per clock cycle, the exact timing specific is determinated by the Clock Settings. And can be seen in the following graphic. In the clock waveform there are two states the 'send' mode and the 'capture' mode. The data lines (MISO, MOSI) cannot change while the clock is on the capture mode. They can only do it while in the 'send' mode.




One thing to have in account is that: As the master is the one that's responsible for keeping the clock signal alive it has to keep sending the clock signal if it won't send data but expects data from the slave.

Once the comunication with that slave has ended the SS is raised again. And then the slave puts the MISO line in high impedance mode.

## $i^2C$

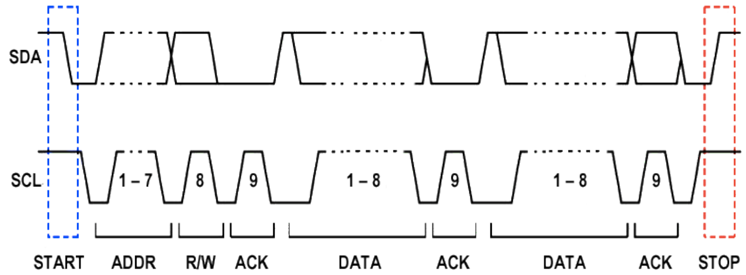
$I^2C$ or IIC stands for Inter-Integrated Circuit, it is a protocol which is synchronous, half-duplex, multi-master and multi-slave. It can address between 127 and 1024 devices but is slower than SPI

This protocol only needs 2 pins

- SCL ⇒ Serial Clock Line
- SDA ⇒ Serial Data Line

### timeline

First the master sends a start bit by pulling down the SDA right before the clock, then sends the device address (7, 8 or 10 bits), a Read/Write bit and wait for the acknowledgment of the slave. Then sends the real data which can be the data address of the device wait for acknowledgment and start again until the stop signal by pulling up the SDA right after the clock.



# Specifications

- 2.4GHz Wi-Fi Connection
- Screen Size of 128 x 64 pixels ($46.5x27.7mm$)
- Endless rotary button
- RGB LED
- Maximum distance to activate the device of 76.2 mm
- 4 x AA Batteries Required
- 3 months autonomy

# So what do we need ?

## A PIC


The choice of the PIC have been done mostly on the power consumption performance and the availability on farnell as well as the number of SPI ports.

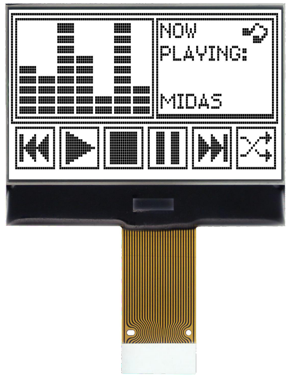
## A Wifi Module


In order to communicate with the server we needed a wifi connection that we found in the atwinc1500, this module is compatible with most wifi encryption and has an SPI interface as well as a low-power mode.

## A Button

We chose this encoder because it fits exactly our needs: it rotates infinitly either way, it has a push button integrated enabling the ability to choose and select within an instant, further to that it also has an RGB led incorporated !

## A screen



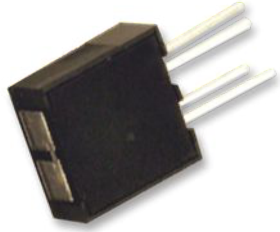

We chose this screen because it is large enough to contain a good list of words,it has an English and Japanese font set, it is bright so we can see it in different environnements and it is 3.3V powered wich make it easy to interface with all other modules.

## Screen Connector

## Memory Socket


As a compusilve buyer you will need a lot of products in your list, that's why we chose to add a memory extension.

## Photo-Diode

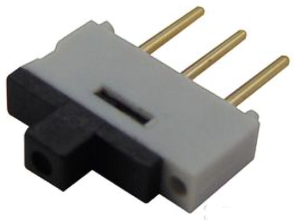


Thanks to the half-god Gregory Le Grand we will use a photo-diode to interact with the device, we chose this one because it can detect a distance of 7.4cm instead of 2.5 for most devices

## Piezo

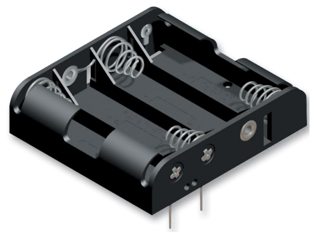
Because everyone loves music, we necessarily wanted a good audio interface that satisfies this requirement.

## Switch



This basic switch will be the one to turn all the lights off, in order to store the product we want a way to switch it completly off keeping the batteries inside

## Battery Holder



Because it wont hold itself...
We chose this one because it widely used, well appreciated and we needed enough mAh to be able to use the device during several months
we have 4 AA batteries holding each one 2850mAh
4 * 2850 = 11400
Our device consumes at most 500mA in case everything is working at the same time
so 11400/500 = 22.8 hours the hard way
considering that we use the device 15 minutes a day we use it 7h45 by month
22 / 7.75 = 2,84 months
In this conditions we can use the device during three months without changing the batteries which is acceptable.

# References and links

We need to be more precise on the consumption, the voltage regulator might not use more than the pic or the screen...
I didn't find a good approximation of the pic consumption
So we have to know on witch frequency the pic will work with the ATWINC and without (idle mode and sleep mode)

| component | reference | price(€) | consumption | Frequency | Datasheets |
|---|---|---|---|---|---|
| SoC | PIC32MX174F256B-I/SO | 3.48 | $\sim$ 200mA @ 3.3V = 660mW | 72Mhz | |
| Wi-Fi | ATWINC1500 | 6.61 | 70mA / 172mA @ 3.3V = 564mW | 26Mhz | |
| Screen | MCCOG128064B12W | 8.69 | 40mA @ 3.3V = 132mW | 64hz | |
| Screen connector | 28FLZ-RSM2-TB(LF)(SN)(P) | 1.37 | | - | |
| Rotary Encoder | PEL12T-4225S-S1024 COM-10597 | 1.88 | 10mA @ 5V = 50mW | - | |
| Photo-Diode | OPB732 | 3.2 | 50mA @ 3.3V = 165mW | - | |
| Buzzer | ABT-407-RC | 1.58 | | - | |
| Voltage Regulator 3.3V | TPS561208DDCT | 0.737 | | - | |
| Micro SD socket | DM3AT-SF-PEJM5(40) | 2,64 | | - | |
| Battery Holder | 2477 | 1.41 | | - | |
| Switch | 09-03290.01 | 1.05 | | - | |
| total | | 32.647 | | | |