Retro-Comm

Introduction

- My goal with this project is to make an **old rotary phone** work again
- The phone should connect to a modern telephone netowrk and be used reliably as a communication device
- It will have all of its **old features** working:
  1. rotary disc for selecting the phone number to be called
  2. ringing bells for incoming calls
  3. receiver being lifted to answer a call / initiate a call
- As well as some **new features** to give it a modern twist:
  1. ability to save important numbers in memory for faster calls
  2. custom ringtones
- The main reason why I chose this project was that it offered me the posibility to **reverse engineer** an interesting old system while also applying the principles learnt during the semester at PM
- The ideea came from the similar project implemeted last year at the PM Fair, which I really liked and I hope I can improve, but also from a few other online sources mentioned in the bibliography section

General Description

The phone works as follows:

- Whenever the receiver is placed in the stand the phone is in settings mode, when the user can change the ringtone, add new phone numbers into memory or call saved numbers, all using the disc for selection
- When the receiver is taken away from the stand the user can select a number to be called using the disc. The arduino will read the disc signals, will construct the number in a string and then pass it via USART to the SIM800L module using the AT command for calling (ATD+number). If the reciever is placed in the stand during the call, the arduino will send the hang up command (ATH)
- When a call is incoming, the RING line from the SIM800L will be set LOW, indicating to the arduino that someone is calling. The arduino will then ring the bells using an electromagnet powered by the MOSFET module, acording to the current ringtone
- Mic and speaker are handled by the SIM800L
- Supply voltage (12V) will be lowered using MP1584EN modules to 4V for the SIM800L and 5V for the

arduino

Hardware Design

Components

Component	Role	Quantity
Arduino Pro Micro	Controlls all other components	1
SIM800L	2G communication	1
MOSFET module	Allow the 12V supply to connect to electromagnet	1
Electromagent	Pull the hammer rod that will hit the bells	1
MP1584EN module	Step down the 12V voltage for SIM800L and Arduino	2
10 kOhms resistor	Voltage divider for the SIM800L	3
1000 uF capacitor	Regulate current for SIM800L during 2A busrsts	1
Electret microphone	Record the voice of the user	1
1N4007 diode	Stop the reverse voltage from the elctromagnet	1
Perfboard	Base used to solder the components	2

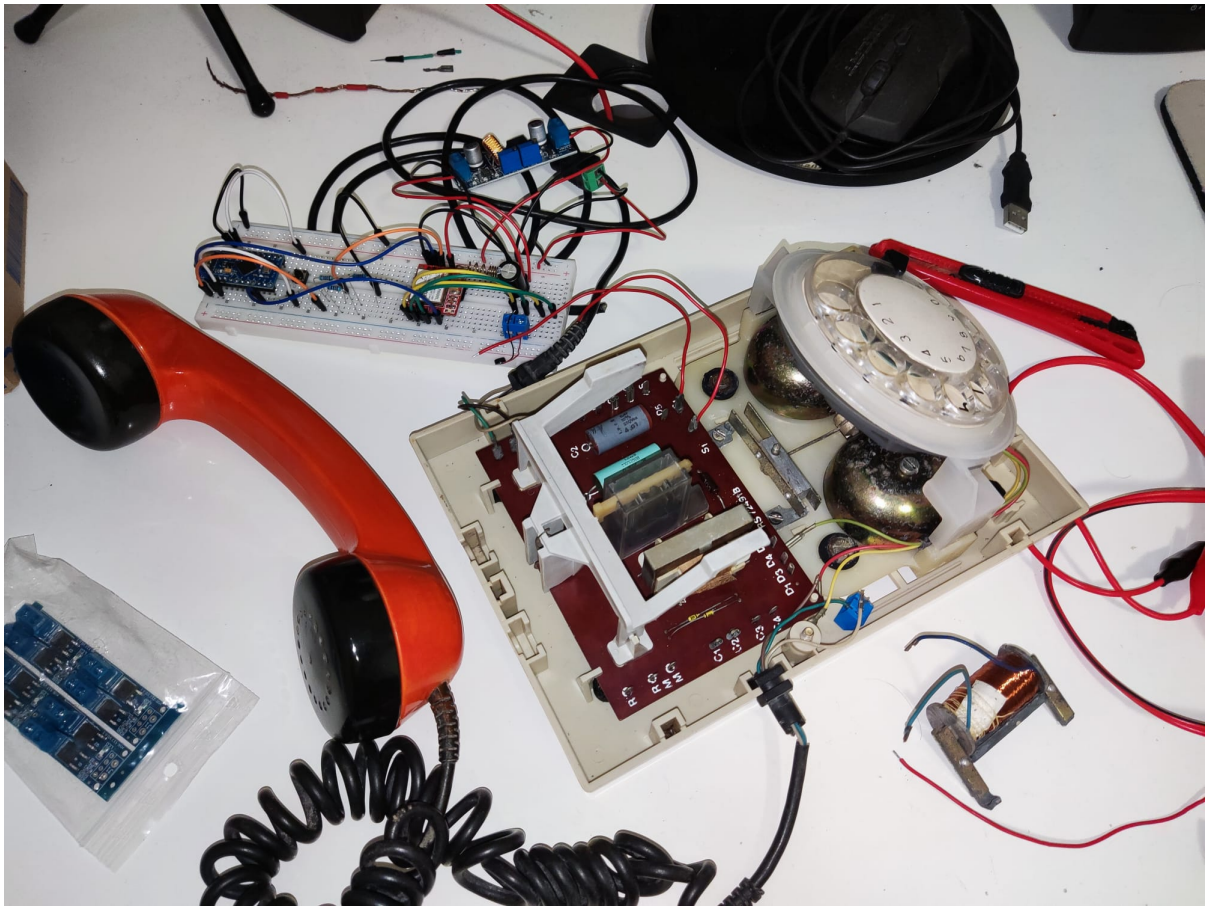
Schematic

Pins used / connections

- The rotary disc encoder is connected to pins PD1 and PD0 in order to use the INT0 and INT1 interrupts to detect the short pulses genrated by the disc for each digit
- The PWM of the MOSFET module is connected to PB6, a pin capable of generating the PWM signal necessary for the ringtone
- The receiver stand, which will basically act just like a button, will be detected from PB5 using a debouncing method
- The RX0 is connected to the TX pin of the SIM800L and the TX0 pin to the RX one of the sim module via a voltage divider

Current status

- I have managed to initiate and receive phone calls using the module. I have connected the old speaker of the phone to the sim module and it seems to work fine. I have also tested the audio with an electret microphone and it works. This means that the most important part of the hardware stage is working
- I still need to find a good way to place the micropone in the phone receiver and deal with the common ground layout used in the telephone in order to be able to use both speaker and microphone togheter
- I have found the connections responsible to detect whether or not the receiver is placed in its stand and will connect them to the arduino. Similarly with the rotary encoder
- Here is a picture of the mess:



- For a proof of the working phone calls I have attached a video on the Moodle assignment Milestone 2 (Hardware)

Software Design

Descrierea codului aplicaţiei (firmware):

- mediu de dezvoltare (if any) (e.g. AVR Studio, CodeVisionAVR)
- librării şi surse 3rd-party (e.g. Procyon AVRlib)
- algoritmi şi structuri pe care plănuiţi să le implementaţi
- (etapa 3) surse şi funcţii implementate

Rezultate Obţinute

Care au fost rezultatele obţinute în urma realizării proiectului vostru.

Concluzii

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună .

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF