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

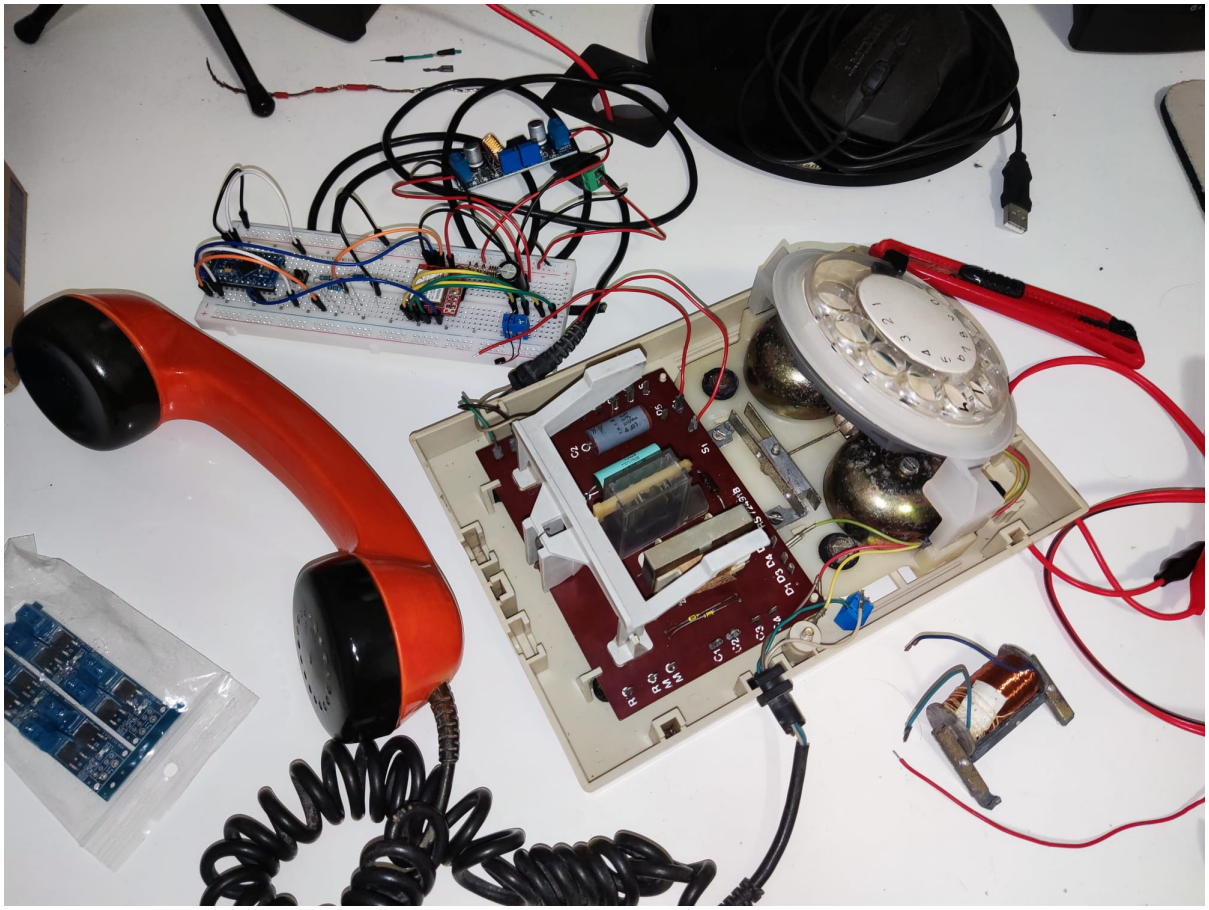
Schematic

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 PB1 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:



Software Design

The code can be found at the following **GitHub Repository**:
https://github.com/Cata1022/Arduino-Rotary-Phone

Software Description

The main parts of the code and the way they interact are:

- INT0 and INT1 interrupts used to count the digit being dialed by monitoring the changes in voltage generated by the disc on the associated pins. An array of the phone number or command being dialed is stored in these interrupts
- Timer1 counts the milliseconds used as a clock for the whole system and is used to count the precise time (3ms) during which the electromagnet is being powered in order to give it just the right impulse (any more than 3ms could overheat the electromagnet, so using an interrupt for this was essential)
- PCINT0 on PB1 is used to detect wheter or not the receiver is placed in its stand. A varaible keeps track of its state, changing the way the phone is configured. When it is placed down, calls can be received, settings like ringtone or favourite numbers can be changed, whereas when it is up a phone number can de dialed and called
- The UART functions ensure the communication between the Arduino and the SIM800L in order to initiale calls (ATD+<phone_number>), answer (ATA), hang up (ATH) or set initial conditions like mic/spk volume, etc.

- The diffrent ringtones are stored in Flash using PROGMEM to not use SRAM unecessarly. A ringtone is represented as an array of “pauses”. That is the time between electromagnet impulses which move the hammer that hits teh bell, making a sound. This way, a song can be represented simply as just the time between its main beats. A simple python script was used to convert a few RTTTL formated songs in the format mentioned above. The selection of the current ringtone being used will be saved, just as the favourite phone numbers, in the EEPROM of the Arduino
- The main loop:

1. monitors what the sim module sends in order to determine if the phone should be ringing (as long as it receives thee RING message) or the phone call ended (NO CARRIER message)
2. checks if a phone number has been fully dialed and the calls it giving the ATD+<phone_number> command
3. ends calls when the receiver is placed back in the stand
4. answers calls when the receiver is lifted up
5. plays the ringtone for as long as RING messages are coming from the sim module
6. processes commands from the user (save a new favourite number, call a saved number, change ringtone)

Libraries used / framework

- PlatformIO was used for compiling and uploading the code to the Arduino
- I used the standard avr libraries like io.h, interrupt.h, as well as a custom simple UART library and string.h for basic string manipulation
- Only in the testing phase I used the Arduino.h libaray in order to make use of its serial library used to comunicate via the integrated USB connection to the PC for debugging purposes. Once the debug is done, I will remove the library as it will not be needed

What is diffrent about this project?

I have managed to improve on the design that inspired me, mentioned in the introduction, by making the old bell of the phone to work again, creating a more authentic retro experience. I also added 2 new features to the phone: custom ringtones + storing favourite phone numbers for fast calling

Labs used

The project uses knowledge learned during the PM labs, primarily from the following labs:

1. **Interrupts**
2. **Timers**
3. **UART**
4. **GPIO**

Results

A demo showing the phone receiving and initiating phone calls can be seen here: https://drive.google.com/file/d/1bky5QWDyqzlO49M3sR-WaWsKY6L0Qj9e/view?usp=sharing

Explanation: the white plastic U-shaped stand is where the reciever will be placed once I put the

cover back on. In the video I use my finger to simulate the weight of the receiver being placed on it. It can be observed that the stand is used to end the call as well as answering it when the bells are ringing. In the end I talked in the microphone and showed that the sound can be heard on the other side, demonstrating that the audio is working (the best way I could on video at least)

Video showing the custom ringtone feature:
https://drive.google.com/file/d/1EGCo3zd8fODHPQl7_8CeNUZMR8lepJmO/view?usp=sharing

Final stage:

Conclusion

It was a challenging but really rewarding experince working on this project. It certainly was one of the most intersting projects I have made for a course. It sparked my interest in the subject, I will surely work on other embedded projects in the future. I have learned quite a lot about microcontroller programming as well as electronics and satisfied my curiosity to understand how a system works and then repurpose it my own way

Resources

- ATmega32U4 Datasheet: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf
- Arduino Pro Micro pinout:

https://cdn.sparkfun.com/datasheets/Dev/Arduino/Boards/ProMicro16MHzv1.pdf
- SIM800L Datasheet: https://www.alldatasheet.com/datasheet-pdf/view/1741389/SIMCOM/SIM800L.html
- Inspiration:

1. https://ocw.cs.pub.ro/courses/pm/prj2025/ccristi/alexandru.beiu
2. Getting Old Rotary Phone To Ring: https://youtu.be/5VrxtBEDkn4?si=HW0godYVfUgG7dM-

Export to PDF