ARDUINO ESP8266 BASED TELEPHONE CALLER ID SYSTEM WITH ANTI-SPAM FUNCTION

Version 1.01 09-Feb-2018

1 INTRODUCTION

This document describes the design and construction of a system based on an Arduino ESP8266 for interpreting and displaying details of incoming telephone calls to fixed telephone network [pots] subscribers. It includes a function to suppress the ring tone for numbers which are either not known or fail other defined criteria (anti-spam function).

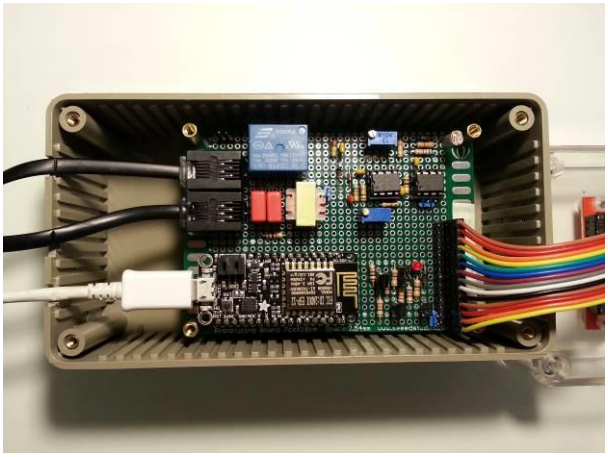
It is compatible with telephone networks which use the AFSK standard for encoding call data. (see the ETSI documents in the reading list)

The core of the solution is an AFSK demodulator written in C/C++ and does not rely on any purpose designed signal processing ICs. It is designed to use standard, easily available components. Additional software components includes a parser, display handler etc. The inbuilt analog to digital converter (ADC) on the ESP8266 is, however, has insufficient performance for this and an MCP3002 is used instead.

The system uses an external server for storage of call history and address book, but can, however, operate in offline mode and display incoming call data but then without matching calls with the address book and the anti spam function is restricted. Call history can also be viewed through a standard web browser using the standard MySQL administration system (or custom php/html pages – not supplied).

This project is not ideal for beginners because it will not be easy to get help if something goes wrong because there is a lot of dependencies on the local environment.

The container showing a window cut in it to enable operation of the touch screen.
The colour red indicates the number was identified as spam.
The (OK) indicator shows the check digit test for data transmission accuracy was successful.
The button **conf** shows the configuration menus.



The container opened, showing the circuit built on a 10 cm x 7 cm prototype board. All components are visible because the reverse side has been reserved for connections.

General disclaimer:

1. Before connecting anything, including this device, to your telephone network, be sure that it does not conflict with any local requirements or requirements of your telephone service provider regarding construction, materials, electrical safety, emission of electromagnetic radiation, certification or other such matters.
2. This development has been tested only in a Swiss network. Although the standards are in principle international, there are differences which may impact the correct functioning in other environments.

2 CONTENTS

2.1 ACKNOWLEDGEMENTS

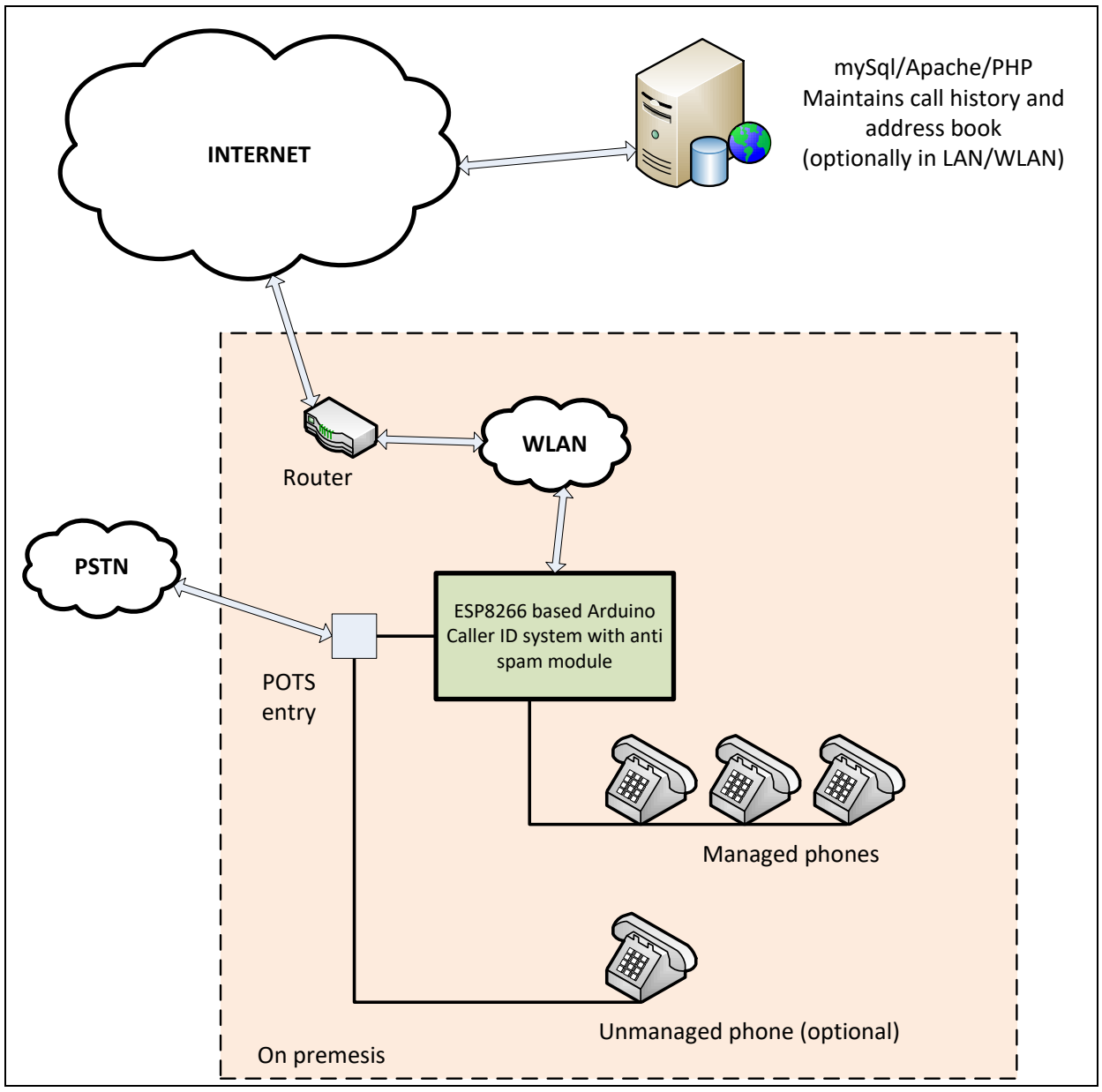
- The software AFSK demodulator is adapted from an APRS modem designed by markqvist (www.unsigned.io)

2.2 ABBREVIATIONS

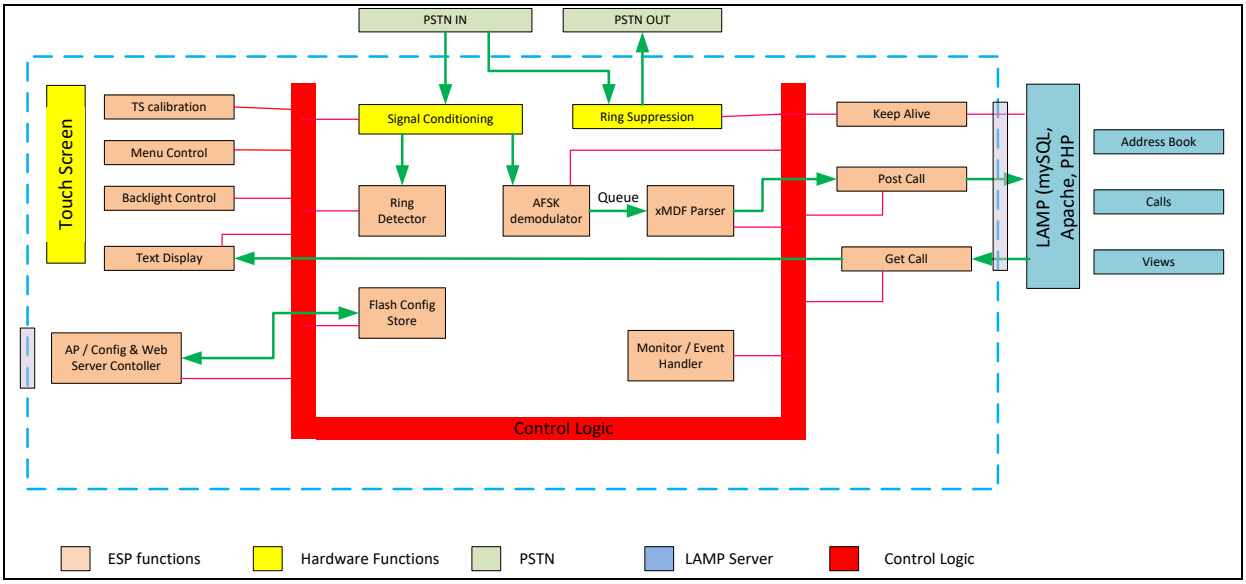
ADC	Analog to digital converter
AP Mode	Access point mode. The ESP8266 creates an access point which wireless devices can connect to. It is used for an initial configuration.
APRS	Automatic Packet Reporting System
AFSK	Audio Frequency Shift Keying
CLIP .	Caller Line Information Presentation
E164	Telephone number representation e.g. +442071838750
Eeprom	Electrically erasable read only memory
ESP8266	A microcomputer architecture compatible with Arduino.
ETSI	European Telecommunications Standards Institute
IC	Integrated Circuit
MCU	Microcontroller Unit
MDMF	Multiple Data Message Format
POTS	Plain Old Telephony System
PSTN	Public Switched Telephone Network
SDMF	Single Data Message Format (not currently supported here)
xAMP	A server architecture consisting of a mySql database, Apache web server and a PHP as a server side scripting language.

3 SOLUTION DESCRIPTION

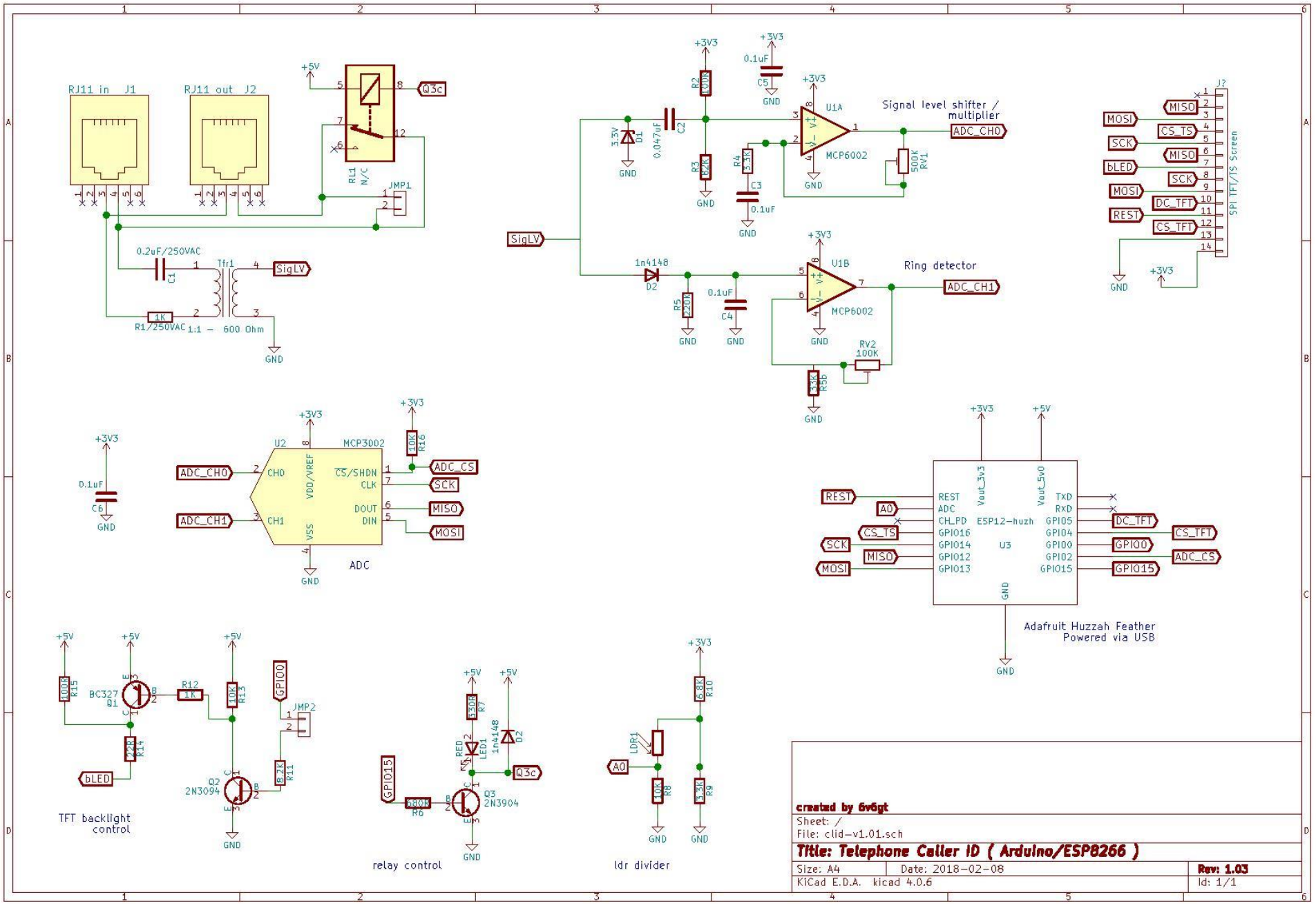
3.1 BIG PICTURE



3.2 FUNCTION BLOCK DIAGRAM



3.3 SCHEMATIC DIAGRAM


RJ11 in J1
RJ11 out J2
+5V
Q3c
R11 N/C
JMP1
0.2uF/250VAC
C1
Tfr1
SigLV
1K
R1/250VAC
1:1 – 600 Ohm
GND
+3V3
0.1uF
C5
R2 100K
3.3V
D1
0.047uF
C2
R3 82K
R4 3.3K
C3 0.1uF
U1A
MCP6002
Signal level shifter / multiplier
ADC_CH0
500K RV1
1n4148
D2
R5 220K
C4 0.1uF
U1B
Ring detector
ADC_CH1
Rv2 100K
R5b 3.3K
J?
SPI TFT/TS Screen
MISO
MOSI
SCK
bLED
CS_TS
DC_TFT
REST
CS_TFT
U2
MCP3002
CH0
CH1
VDD/VREF
VSS
CS/SHDN
CLK
DOUT
DIN
ADC_CS
SCK
MISO
MOSI
R16 10K
C6 0.1uF
ADC
U3
ESP12-huzh
Vout_3v3
Vout_5v0
REST
ADC
CH_PD
GPIO16
GPIO14
GPIO12
GPIO13
TxD
RxD
GPIO5
GPIO4
GPIO0
GPIO2
GPIO15
A0
Adafruit Huzzah Feather
Powered via USB
R15 100K
BC327 Q1
R12 1K
R13 10K
R14 22K
bLED
GPIO0
JMP2
Q2 2N3094
R11 8.2K
TFT backlight control
GPIO15
R6 330K
R7 330R
RED LED1
1n4148 D2
Q3 2N3904
relay control
LDR1
R10 6.8K
R8 10K
R9 3.3K
ldr divider
created by 6v6gt
Sheet: /
File: clid-v1.01.sch
Title: Telephone Caller ID (Arduino/ESP8266)
Size: A4
Date: 2018-02-08
Rev: 1.03
KiCad E.D.A. kicad 4.0.6
Id: 1/1

4 FEATURE LIST

Display of incoming call information including number, timestamp and subscriber name when supplied by the telephone network, supplemented by address book information from the xAMP server for registered callers.

Ring detection and display of ringing status.

Interface to an xAMP server (MySQL/apache/PHP) for call storage and retrieval with address book maintenance.

Spam defence with multiple options including white list call authorization and white list to reject calls. Call rejection is implemented by ring suppression resulting in spam calls being routed to voice mail after a preset time.

Call history can be viewed either through the device, with paging through using the touch screen or it can be viewed online direct in the mySql database through predefined screens.

Display backlight intensity is 2 level dependent on ambient lighting and with flicker suppression.

New call indicator which is cancelled by any touch screen operation.

Software features to assist debugging of signal quality related problems.

Failsafe design. During periods of any malfunction (power failure, system crash etc.), call ringing is not suppressed. The suppression is activated by an n/c relay which is opened only after a positive indication of an unauthorized call attempt. It is strongly advised to have a voice mail system to accept any calls identified as spam or otherwise unanswered.

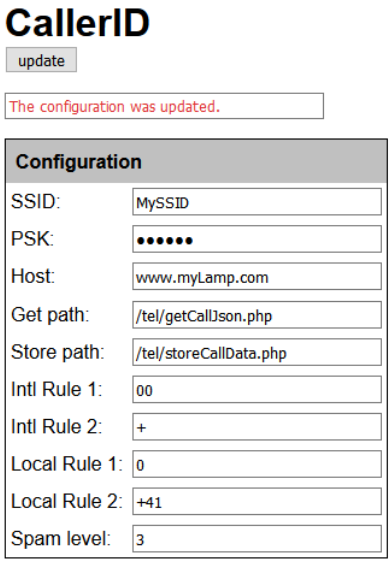
Selection between online and offline mode. In offline mode, there is no storage of call data and no spam suppression. Selection of offline mode is through a touch screen configuration option. Configuration options relevant to online mode are set through a web browser.

The touch screen has a 2 point calibration routine which requires the user to mark 2 points on the screen and it then calculates the mapping between the touch pad and the screen and stores the settings in flash memory.

A light weight and simple to use navigation bar class has been developed for handling paging, configuration options etc.

Note: currently only MDMF format CLIP is supported here.

5 WEB CONFIGURATION



This is used for online relevant settings. This includes the local wireless LAN credentials and the location of the server where call data is stored and retrieved.

The initial entry of the WLAN credentials is made by using the configuration option to put the device into AP mode. The screen instructions indicate the SSID of the connection and the IP address of the temporary server.

Intl Rule is for converting international number to E164 format. In this example, a leading instance of 00 is replaced with a +.

Local rule is to convert local numbers to E164 format. In this example, a leading instance of 0 is replaced with +41.

The spam level is 0 for no action, 1 for rejection of suppressed numbers, 2 for rejection of numbers without a network name and 3 rejection of all calls that are not from named callers in the address book (white list screening).

6 LOCAL CONFIGURATION

This is managed through a touch screen interface. The hierarchy is described here.

1. Set Debug Options

1.1 Chose 0 to 3. Debug information is written to the serial console. Generally 0 is minimal debugging information and 3 has most. Option 1 is special in that the output of the demodulator

is written directly to the screen in a formatted form and is used for configuring the signal conditioner potentiometer.

2. Set Online / Offline Offline is the default initial configuration.

3. Forces the device into AP (Access Point) mode to allow browser configuration of the online mode relevant options

4. View selected configuration settings (such as software version, online/offline mode etc.)

5. Restart the system (power fail restart).

6. Exit configuration mode.

7 xAMP Server Configuration

This is to store and retrieve the call data and to maintain the address book.

No instructions appear here about the general aspects of configuring a xAMP server. Only the application specific objects are mentioned.

The MySQL database has two tables and a view. These are described below and have to be set up manually.

In addition , three PHP files are required and these are in the deployment pack. One for storing calls, one for retrieving stored calls and one for access credentials.

7.1 Table: calls

To be named “calls” (case sensitive)

Field	Type	Null	Key	Default	Extra
id	mediumint(9)	NO	PRI	*NULL*	auto_increment
timeStampSystem	timestamp	NO		CURRENT_TIMESTAMP	
telNo	varchar(15)	YES		*NULL*	
nameNet	varchar(20)	YES		*NULL*	
nameTelBook	varchar(20)	YES		*NULL*	
dateStampNet	varchar(12)	YES		*NULL*	
checkDigitOk	tinyint(4)	NO		*NULL*	
runNumber	int(11)	NO		*NULL*	
numberInRun	int(11)	NO		*NULL*	

7.2 TABLE: TELBOOK

To be named “telBook” (case sensitive)

Field	Type	Null	Key	Default	Extra
telNo	char(15)	NO	PRI		
name	varchar(20)	YES		*NULL*	

7.3 VIEW: CALLEXP

To be named “callExp” (case sensitive)

Field	Type	Null	Key	Default	Extra
id	mediumint(9)	NO		0	
timeStampSystem	timestamp	NO		0000-00-00 00:00:00	
telNo	varchar(15)	YES		*NULL*	
nameNet	varchar(20)	YES		*NULL*	
dateStampNet	varchar(12)	YES		*NULL*	
checkDigitOk	tinyint(4)	NO		*NULL*	
numberInRun	int(11)	NO		*NULL*	
runNumber	int(11)	NO		*NULL*	
name	varchar(20)	YES		*NULL*	

This is an open join of tables **calls** and **telBook** over column **telNo** which effectively adds the column **name** to the columns in **calls** and, if available, populates it.

```
CREATE ALGORITHM=UNDEFINED DEFINER=`myUser`@`%` SQL SECURITY DEFINER VIEW
`callExp` AS select `t1`.`id` AS `id`,`t1`.`timeStampSystem` AS
`timeStampSystem`,`t1`.`telNo` AS `telNo`,`t1`.`nameNet` AS
`nameNet`,`t1`.`dateStampNet` AS `dateStampNet`,`t1`.`checkDigitOk` AS
`checkDigitOk`,`t1`.`numberInRun` AS `numberInRun`,`t1`.`runNumber` AS
`runNumber`,`t2`.`name` AS `name` from (`calls` `t1` left join `telBook`
`t2` on((`t1`.`telNo` = `t2`.`telNo`)))
```

8 SOME DETAILS OF DESIGN DECISIONS

8.1 HARDWARE

The ESP8266 is an Adafruit Huzzah Feather. The built in USB/UART interface is useful also for loading software to system when it is in an enclosure.

The ADC is an MCP3002 SPI which handles easily the 9600 Hz demodulator (and ring detector). This has only 2 analog inputs. Better would have been a 4 or 8 pin option to compensate for the limited amount of pins available on the ESP8266.

The signal conditioner uses an MCP6002 op amp (rail to rail) which handles the AFSK and the ring detector. The ring detector is a special design which detects a 16 to 70 Hz signal above 0.7 volts. This avoids having any further connections, apart from the isolating transformer, to the high voltage side of the telephone network. The frequency detection is done in software.

The touch screen is a 2.8 inch TFT SPI using an XPT2046 controller (model TJCTM24028-SPI). I have had two various types from the same retailer, one requiring some modifications to XPT2046 library to get a sufficiently stable screen position from it. A change of screen type could have a system wide impact.

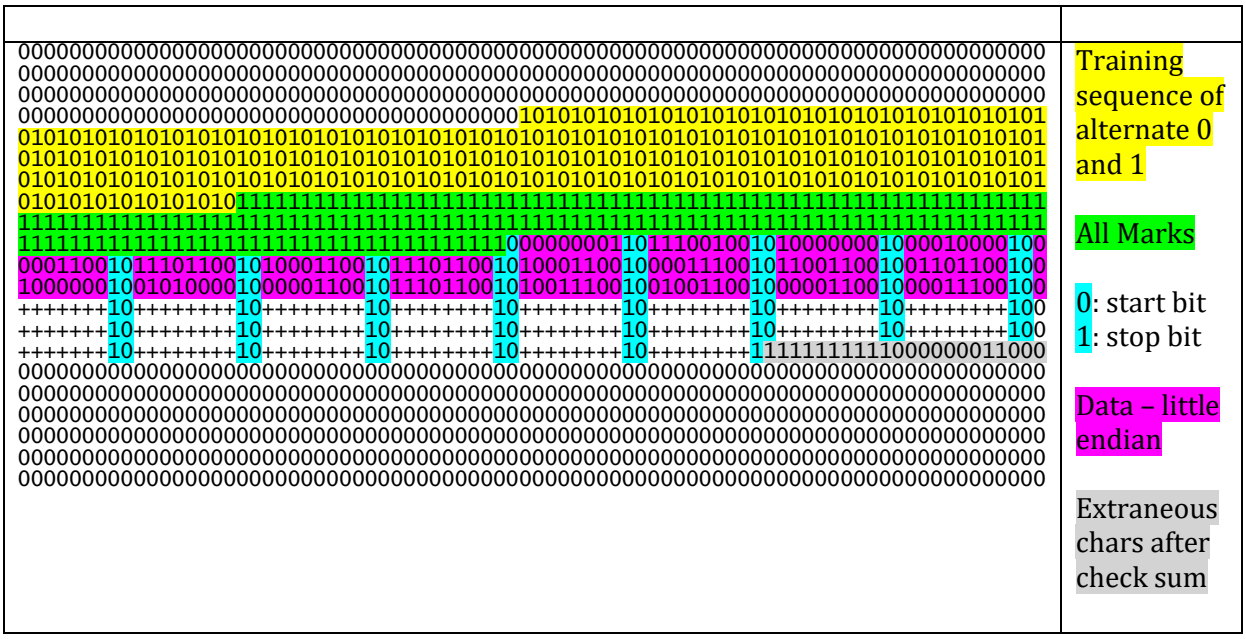
8.2 SOFTWARE

The AFSK routine taken from the markqvist modem has, apart from the removal of the modulator, been left relatively untouched including comments. The ring detector has been loosely attached to the demodulator because both use the same ADC and are both controlled via timer1.

A monitor has been implemented to handle events such as timeout after being left in configuration mode, no connection to the server etc. The default action is to retrieve the latest call from the database and display it. A restart is forced if an error condition persists.

9 DATA FORMATS

The best way of describing these is by example, using the output from the application in debug mode 1 which produces an 80 bit wide dump of the signal direct from the demodulator.



The fields between the start(0) and stop(1) marks, some highlighted in purple are the actual call meta data and the first few are decoded below. The +++++++ pattern represents further characters in the data stream which have been obscured here. The parser is fed these bytes and produces the output for the display and storage in Eeprom.

Data Stream Value (little endian)	Hex Value / character	Description
00000001	0x80	Header byte to indicate MDMF data format
11100100	0x27	Length of following data packet in bytes
10000000	0x01	Date parameter type header marker
00010000	0x08	Length of date parameter in bytes
00001100	0x30	Ascii Digit 0 (first digit of month)
11101100	0x37	Ascii Digit 7 (second digit of month. 7 = July)
10001100	0x31	Ascii Digit 1 (first digit of day of month)
11101100	0x37	Ascii Digit 7 (second digit of day of month – so the 17th)
Etc. Etc.		

The encoding is described in several of the quoted references. [Ref5] and [Ref6] are the easiest to follow. The ETSI documents [Ref7] and [Ref8] are the most comprehensive.

10 TROUBLE SHOOTING TIPS.

1. Keep any mobile phone you are using for testing a few meters away from the circuit to avoid interference.
2. If you don't get a clean pattern in debug mode 1 on the serial console when there is no activity on the line, i.e. no ringing, then you have to solve that before proceeding further. The variable resistor RV1 should have an initial setting of around 60k, but adjust as required. Debug is mode 1 selectable via the touch screen.
3. The ring detector should be tuned via the potentiometer so the mark/space ratio is about 50 % and a special sketch may help to make this ratio visible (not supplied). An oscilloscope could also be useful. The ring should be a sine wave, but in reality may be nothing like one. In the worst case, the circuit may have to be tuned to local conditions.
4. In standard mode, debugging information is written anyway to the serial console which can be useful if the signal delivered to the MCU is of a reasonable quality.
5. There is some fine tuning of the mark and space frequencies available to handle different network standards. See AFSK.h (source software)
6. The Adafruit Huzzah feather is sensitive to what is connected to GPIO #0 which can interfere with the serial monitor. Using it as an output pin through 8.2K resistor seemed OK. For safety, a jumper appears on the design to allow GPIO #0 to be isolated.
7. Care should be taken that no code changes allow the various SPI devices to interfere with each other. The SPI ADC is controlled via timer1 and the SPI screen components are controlled via the loop and these are not synchronous.
8. I had some success using an audio recording of an AFSK signal captured from a telephone line, and feeding this into the demodulator via the PC sound card for testing.

11 A DOCUMENT COLLECTION, SOME DIRECTLY REFERRED TO ABOVE OR USED IN THE SOLUTION AND SOME GENERALLY RELEVANT FURTHER READING.

[Ref1] LibAPRS an Arduino soft modem library which provided the basis for the demodulator code.

http://unsigned.io/projects/libaprs/

[Ref2] Rane Note: Interfacing Audio and Pots including some basic electrical parameters of the telephone network and an interface circuit.

http://www.rane.com/note150.html

[Ref3] Randolf Telecom Inc. AN-4 (Midcom TN #98) LOW COST TELEPHONE LINE INTERFACE (DAA, FXO)

This is a comprehensive description including a specimen circuit for interfacing to POTS telephone networks

http://www.randolph-telecom.com/articles/AN-4,%20Low%20cost%20telephone%20line%20interface%20_DAA,%20FXO_.pdf

[Ref4] Cypress AN2336 - PSoC® 1 - Simplified FSK Detection

This is an example of using a Cypress manufactured chip for FSK signal processing. It is interesting here because it contains a comprehensive theoretical description of the processing techniques which are relevant to the AFSK demodulator in this solution.

http://www.cypress.com/documentation/application-notes/an2336-psoc-1-simplified-fsk-detection

[Ref5] holtek Type I caller ID using the HT9032

http://www.holtek.com.tw/documents/10179/116745/an0053e.pdf

[Ref6] EXAR TAN008 Designing Caller Identification Delivery Using XR-2211 For U.S.

https://www.exar.com/files/documents/tan_008.pdf

The two references above are similar application notes for obsolete ICs which describe the data formats and parsing of Caller Line Identification information. Exar has also a similar document relevant to British Telecom networks (TAN009)

ETSI documents

[Ref7] **ETSI EN 300 659-3 V1.3.1**
http://www.etsi.org/deliver/etsi_en/300600_300699/30065903/01.03.01_40/en_30065903v010301o.pdf

[Ref8] **ETSI 2 EN 300 659-1 V1.3.1**
http://www.etsi.org/deliver/etsi_en/300600_300699/30065901/01.03.01_60/en_30065901v010301p.pdf

These are very comprehensive documents describing and specifying relevant telephony data formats and encoding standards at various layers.

[Ref9] arduino caller line identification system

https://forum.arduino.cc/index.php?topic=490392.0

This is a previous development but without an anti-spam function but with local storage of call data.

12 SOFTWARE DISTRIBUTION / DEPLOYMENT PACK

This a zip archive containing the ESP8266 software (multiple components) and 3 PHP files which are used by the ESP8266 for storing and retrieving call data.

Tested with the following software versions:

- ESP8266 Arduino core 2.4.0
- PHP 5.2.17
- MySQL 5.1.59-log

Location: https://forum.arduino.cc/index.php?board=29.0