arduino, lecteur-son, em

Téléphone BMR

/!\ en cours de rédaction 22 octobre 2019

C'est un superbe téléphone sensuel en forme de bouche qui a servi à diffuser une suite d'entretiens sur le sujet des règles. Il a été utilisé comme borne d'écoute pour la journée/soirée “Bois mes règles” organisée par Gast! à Quimperia / La Baleine en mars 2018.

Le clavier du téléphone permet de déclencher la lecture d'un enregistrement que l'on peut écouter dans le combiné, comme un téléphone quoi!

Le circuit du téléphone est remplacé par un arduino nano qui déclenche la lecture d'un fichier mp3 par un circuit DFPlayer, le haut-parleur utilisé est le haut-parleur original du combiné. Le clavier original du téléphone est aussi réutilisé et l'ensemble est alimenté par un transfo qui récupère le courant du secteur pour en faire un courant 9V continu.

Réutiliser le haut parleur du téléphone

Le combiné est relié au poste par une prise RJ11 (seuls quatre fils sont utilisés) :

- Vert : micro
- Rouge : micro
- Jaune : HP
- Noir : HP

On se servira des fils jaunes et noirs pour relier le haut-parleur au circuit

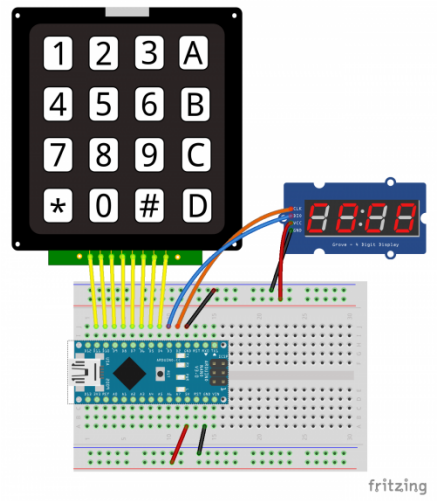
Réutiliser le clavier du téléphone

Chaque touche active une liaison entre deux broches.
Méthode pour retrouver la matrice :

- numéroter les broches ou les nommer
- sur du papier, dessiner une grille avec autant de colonnes et de rangées que sur le clavier et dessiner le clavier
- en utilisant un multimètre pour tester la continuité, tester toutes les combinaisons de broches possibles, en appuyant sur chaque touche, l'une après l'autre
- noter pour chaque touche les 2 broches utilisées

Connexions du clavier sur nappe : R4, R3, R2, R1, VSS, C1, C2, C3, C4, PT

	C1	**C2**	**C3**	**C4**
R1	1	2	3	
R2	4	5	6	FLASH
R3	7	8	9	
R4	*	0	#	REDIAL



telephone_BMR_test_clavier_002.ino (cliquer pour afficher le code)

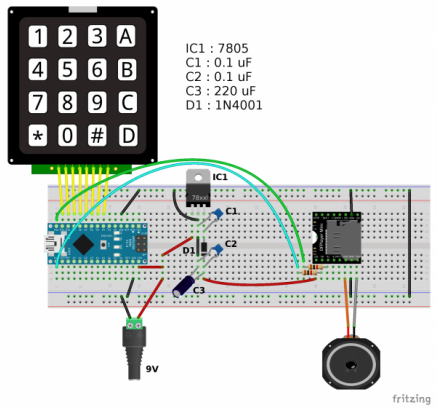
telephone_BMR_test_clavier_002.ino

```
/* deuxième essai avec une autre library pour l'afficheur 7 segment
 *  https://github.com/bremme/arduino-tm1637
 *  SevenSegmentTM1637 de Bram Harmsen
 */

// ***************************************************
// Définitions pour l'afficheur U7 segments à 4 chiffres
#include "SevenSegmentTM1637.h"

const byte PIN_CLK = 2;   // define CLK pin (any digital pin)
const byte PIN_DIO = 3;   // define DIO pin (any digital pin)
SevenSegmentTM1637    display(PIN_CLK, PIN_DIO);

// ***************************************************
// Définitions pour le clavier matrice
#include <Keypad.h>

const byte ROWS = 4; //quatre rangées
const byte COLS = 4; //quatre colonnes
char keys[ROWS][COLS] = {
  {'1','2','3','Z'},
  {'4','5','6','F'},
  {'7','8','9','Z'},
  {'*','0','#','R'}
};

/* Correspondances entres les broches identifiées sur le téléphone et les pins de l'arduino
        R1 : 7       C1 :  8
        R2 : 6       C2 :  9
        R3 : 5       C3 : 10
        R4 : 4       C4 : 11          */
byte rowPins[ROWS] = { 7,  6,  5,  4}; // connecter les rangées à ces pins
byte colPins[COLS] = { 8,  9, 10, 11}; // connecter les colonnes à ces pins


Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup() {

  Serial.begin(9600);
  // Initialiser l'afficheur à 4 chiffres
  display.begin();            // initializes the display
  display.setBacklight(100);  // set the brightness to 100 %
  display.print("INIT");      // display INIT on the display
  delay(1000);                // wait 1000 ms

}

void loop() {

  char key = keypad.getKey();

  if (key){
    display.print(key);
    Serial.println(key);
    delay(300);
  } else {
    display.clear();
  }
}
```

Utiliser un DFPlayer mini

Montage complet

Le montage complet est alimenté en 9V par un transfo externe, dont la tension est transformée en 5V pour alimenter l'arduino. Le fil d'alimentation est enroulé (deux spires) autour d'un anneau de ferrite pour réduire les parasites électromagnétiques qui parasitent le son. L'afficheur à 4 chiffres n'est pas utilisé et la sortie audio se fait sur le haut-parleur du combiné.



telephone_BMR_complet.ino (cliquer pour afficher le code)

telephone_BMR_complet.ino

```
/* Téléphone BMR version 6
 *  SANS affichage 4 x 7 digits
 *  décodage de clavier matriciel
 *  et sortie audio
 *    soit sur sortie jack TRRS, cf. ci-dessous myDFPlayer.enableDAC();
 *    soit sur sortie combiné, avec l'ampli intégré au DFPlayer mini
 *  version 6 : détecte quand même le clavier si pas de connection avec le DFPlayer
 *    Quimper, La Baleine, 6 mars 2018, pierre@lesporteslogiques.net
 */

// ***************************************************
// Définitions pour l'afficheur 7 segments à 4 chiffres
/*
#include "SevenSegmentTM1637.h"

const byte PIN_CLK = 2;   // define CLK pin (any digital pin)
const byte PIN_DIO = 3;   // define DIO pin (any digital pin)
SevenSegmentTM1637    display(PIN_CLK, PIN_DIO);
*/
// ***************************************************
// Définitions pour le clavier matrice
#include <Keypad.h>

const byte ROWS = 4; //quatre rangées
const byte COLS = 4; //quatre colonnes
char keys[ROWS][COLS] = {
  {'1','2','3','Z'},
  {'4','5','6','F'},
  {'7','8','9','Z'},
  {'*','0','#','R'}
};

/* Correspondances entres les broches identifiées sur le téléphone et les pins de l'arduino
        R1 : 7       C1 :  8
        R2 : 6       C2 :  9
        R3 : 5       C3 : 10
        R4 : 4       C4 : 11          */

byte rowPins[ROWS] = { 7,  6,  5,  4}; // connecter les rangées à ces pins
byte colPins[COLS] = { 8,  9, 10, 11}; // connecter les colonnes à ces pins

// ***************************************************
// Définitions pour le lecteur de son

#include "Arduino.h"
#include "SoftwareSerial.h"
#include "DFRobotDFPlayerMini.h"

SoftwareSerial mySoftwareSerial(12, 13); // RX, TX
DFRobotDFPlayerMini myDFPlayer;
void printDetail(uint8_t type, int value);
```

```
int DFPvolume = 18;

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup() {

  Serial.begin(9600);
  // Initialiser l'afficheur à 4 chiffres
  /*
  display.begin();            // initializes the display
  display.setBacklight(100);  // set the brightness to 100 %
  display.print("INIT");      // display INIT on the display
  delay(1000);                // wait 1000 ms
  */

  mySoftwareSerial.begin(9600);
  Serial.println();
  Serial.println(F("DFRobot DFPlayer Mini Demo"));
  Serial.println(F("Initializing DFPlayer ... (May take 3~5 seconds)"));

  if (!myDFPlayer.begin(mySoftwareSerial)) {  //Use softwareSerial to communicate with mp3.
    Serial.println(F("Unable to begin:"));
    Serial.println(F("1.Please recheck the connection!"));
    Serial.println(F("2.Please insert the SD card!"));
    //while(true);
    delay(1000);
    Serial.println(F("Lancement en mode test clavier"));
  } else {
    Serial.println(F("DFPlayer Mini online."));

    myDFPlayer.setTimeOut(500); //Set serial communictaion time out 500ms

    //myDFPlayer.enableDAC(); // pour branchement casque ou sortie ligne

    delay(500);

    myDFPlayer.volume(DFPvolume);  //Set volume value. From 0 to 30

    delay(300);

    Serial.print("nb de fichiers presents dans le repertoire 01 : ");
    Serial.println(myDFPlayer.readFileCountsInFolder(1));
    //myDFPlayer.play(1);  //Play the first mp3
  }
}

void loop() {

  static unsigned long timer = millis();

  char key = keypad.getKey();

  if (key){
    //display.print(key);
    Serial.println(key);
    if (millis() - timer > 1000) {
      playSound(key);
      timer = millis();
      //myDFPlayer.next();  //Play next mp3 every 3 second.
    }
    delay(300);

    if (myDFPlayer.available()) {
      printDetail(myDFPlayer.readType(), myDFPlayer.read()); //Print the detail message from DFPlayer to handle different errors
and states.
    }
  } /*else {
    display.clear();
  }*/
}

void playSound(char key) {

  myDFPlayer.volume(DFPvolume);
  delay(100);

  switch(key) {
    case '0':
      myDFPlayer.playFolder(1, 1); //jouer le mp3 SD:/01/001.mp3; Folder Name(1~99); File Name(1~255)
      break;
    case '1':
      myDFPlayer.playFolder(1, 2);
      break;
    case '2':
      myDFPlayer.playFolder(1, 3);
      break;
    case '3':
      myDFPlayer.playFolder(1, 4);
      break;
    case '4':
      myDFPlayer.playFolder(1, 5);
      break;
    case '5':
      myDFPlayer.playFolder(1, 6);
```

```
        break;
      case '6':
        myDFPlayer.playFolder(1, 7);
        break;
      case '7':
        myDFPlayer.playFolder(1, 8);
        break;
      case '8':
        myDFPlayer.playFolder(1, 9);
        break;
      case '9':
        myDFPlayer.playFolder(1, 10);
        break;
      case '*':
        myDFPlayer.playFolder(1, 11);
        break;
      case '#':
        myDFPlayer.playFolder(1, 12);
        break;
      case 'F':
        myDFPlayer.playFolder(1, 13);
        break;
      case 'R':
        myDFPlayer.playFolder(1, 14);
        break;
    }
}

void printDetail(uint8_t type, int value){
  switch (type) {
    case TimeOut:
      Serial.println(F("Time Out!"));
      break;
    case WrongStack:
      Serial.println(F("Stack Wrong!"));
      break;
    case DFPlayerCardInserted:
      Serial.println(F("Card Inserted!"));
      break;
    case DFPlayerCardRemoved:
      Serial.println(F("Card Removed!"));
      break;
    case DFPlayerCardOnline:
      Serial.println(F("Card Online!"));
      break;
    case DFPlayerPlayFinished:
      Serial.print(F("Number:"));
      Serial.print(value);
      Serial.println(F(" Play Finished!"));
      break;
    case DFPlayerError:
      Serial.print(F("DFPlayerError:"));
      switch (value) {
        case Busy:
          Serial.println(F("Card not found"));
          break;
        case Sleeping:
          Serial.println(F("Sleeping"));
          break;
        case SerialWrongStack:
          Serial.println(F("Get Wrong Stack"));
          break;
        case CheckSumNotMatch:
          Serial.println(F("Check Sum Not Match"));
          break;
        case FileIndexOut:
          Serial.println(F("File Index Out of Bound"));
          break;
        case FileMismatch:
          Serial.println(F("Cannot Find File"));
          break;
        case Advertise:
          Serial.println(F("In Advertise"));
          break;
        default:
          break;
      }
      break;
    default:
      break;
  }
}
```

Sources et ressources

Bibliothèque arduino keypad : http://playground.arduino.cc/Code/Keypad
Bibliothèque pour afficheur 4 digits 7 segments de Bram Harmsen : https://github.com/bremme/arduino-tm1637
Utiliser un clavier matriciel : https://playground.arduino.cc/Main/KeypadTutorial