Bibliothèque Minitel1B pour Arduino (2 versions)



Éric Sérandour

Bidouilleur, auteur de littérature expérimentale, enseignant dans le secondaire.

Site Web :
https://entropie.org

La programmation des microcontrôleurs ATmega avec Arduino m'a ramené quelques décennies en arrière, à l'époque où les micro-ordinateurs 8 bits faisaient rêver l'adolescent que j'étais. À la maison, il n'y avait des ordinateurs que pendant les vacances scolaires, mon père ramenant occasionnellement un ZX81 ou un TO7/70. La seule machine à clavier toujours présente était un Minitel.

Beaucoup de Français ont une histoire personnelle avec le Minitel. J'ai pu m'en rendre compte lorsque j'ai transformé un Minitel en « livre d'or numérique » lors d'un événement au musée d'art et d'histoire de Saint-Brieuc, puis lors d'une « Fête de la Science ». Le public était particulièrement enthousiaste et chacun y allait de ses souvenirs. Le Minitel a la particularité de relier les générations et d'inviter au dialogue : objet de curiosité pour les enfants, il amène les plus âgés à parler de l'arrivée des premiers outils numériques dans les foyers dans les années 1980.

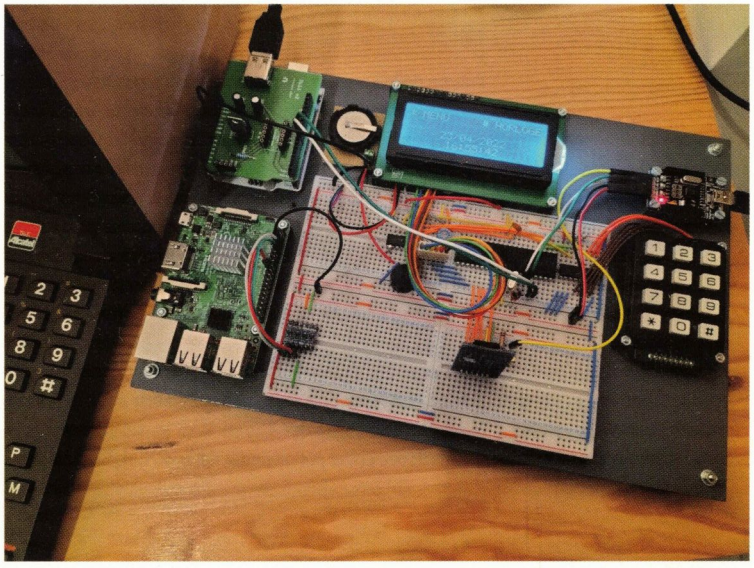
C'est en 2016 que j'ai initié l'écriture de la bibliothèque Minitel1B pour Arduino. Cette bibliothèque existe en deux versions : Minitel1B_Soft et Minitel1B_Hard. Toutes deux sont téléchargeables sur GitHub :
https://github.com/eserandour/Minitel1B_Soft
https://github.com/eserandour/Minitel1B_Hard

Minitel1B_Soft fait appel à SoftwareSerial.h qui permet d'ajouter un port série logiciel à un microcontrôleur du type ATmega328P, celui qui équipe la carte Arduino Uno. Dans ce cas, le port série matériel est conservé pour communiquer avec l'ordinateur et les broches indépendantes définies comme port série logiciel permettent d'échanger avec le Minitel.

Minitel1B_Hard est indépendante de SoftwareSerial.h et permet de faire communiquer le Minitel et un microcontrôleur par l'intermédiaire d'un de ses ports séries matériels. Personnellement j'utilise un ATmega1284P. J'ai été amené à développer cette nouvelle bibliothèque parce qu'assez vite la carte Arduino Uno a montré ses limites au niveau mémoire. Certains bugs apparaissaient dès que les programmes devenaient gourmands en données textuelles par exemple. Je me suis donc tourné vers l'ATmega1284P qui dispose de deux ports séries matériels, de 16 ko de RAM et de 128 ko de mémoire Flash (à comparer aux 2 ko de RAM et 32 ko de mémoire Flash de l'ATmega328P). Une autre alternative intéressante est d'utiliser la carte Arduino Mega 2560 (plusieurs ports séries matériels, 8 ko de RAM et 256 ko de mémoire Flash). Lomig Perrotin, le développeur du projet Zardos, utilise ma bibliothèque en basant son projet sur une carte Arduino Mega. De son côté, Iodeo développe un projet Minitel sur une base ESP32. Il a d'ailleurs contribué à quelques améliorations concernant la bibliothèque.

Comment relier un Minitel et une carte Arduino ?

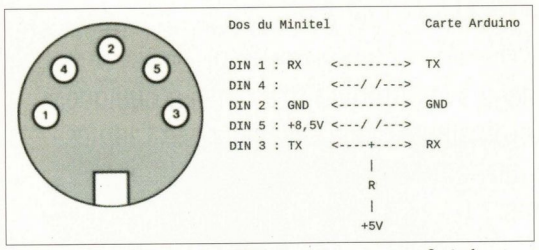
Dans la bibliothèque, vous trouverez un fichier README.txt qui indique les branchements à effectuer entre la prise DIN qui se trouve à l'arrière des Minitel 1B (prise péri-informatique 5 contacts à 45°) et la carte Arduino (voir schéma ci-dessous). Un élément très important, auquel je n'avais pas prêté attention au départ, concerne la sortie TX du Minitel : celle-ci est à collecteur ouvert, elle nécessite donc une résistance de tirage (pull-up). Personnellement, j'utilise une résistance de 10 kilohms. Sans cette résistance R, la communication ne pourra pas se faire correctement. Je récupère le +5V par l'intermédiaire de la broche DIN 5 (13 V à vide sur mon Minitel) sur laquelle je branche un régulateur 5V. Ceci permet d'alimenter la carte Arduino directement avec le Minitel. À noter que tous les Minitels ne permettent pas d'exploiter cette 5e broche (cette fonction n'est pas disponible sur les versions dont l'identification porte les références Cu2 à Cu4 incluses ; elle est par contre disponible pour les versions Cu5 / Cu; / Cu< / Bu0 et suivants).
Pour la fabrication du câble, je vous invite à aller voir sur mon site Web à cette adresse :
https://entropie.org/3615/index.php/hardware/.



Ma plate-forme de travail à base d'ATmega 1284P, connectée au Minitel.

Présentation de la bibliothèque Minitel1B

Pour écrire cette bibliothèque, j'ai utilisé un document de 193



pages édité par les PTT de l'époque : « Spécifications Techniques d'Utilisation du Minitel 1B », téléchargeable aujourd'hui à cette adresse : https://www.goto10.fr/minitel/specifications/stum1b.pdf

Ce document date de novembre 1986 et décrit le fonctionnement du Minitel. La lecture en a été pour moi passionnante. Elle m'a permis de rentrer dans l'intimité du Minitel et lorsque j'ai commencé à écrire la bibliothèque, j'ai tout de suite voulu coller au plus près de ce document. Mon code source renvoie d'ailleurs continuellement à ses pages.

Hello World !

Une étape importante qui fera adopter un outil plutôt qu'un autre est le fameux « Hello World ! ». C'est donc par son entremise que je vous invite à essayer la bibliothèque Minitel1B, dans une de ses deux versions. Au préalable, il faudra télécharger la bibliothèque au format ZIP depuis GitHub (Minitel1B_Soft-master.zip ou Minitel1B_Hard-master.zip) et la décompresser dans votre répertoire sketchbook/libraries/, sketchbook étant l'emplacement du carnet de croquis dans le logiciel Arduino. Le fichier HelloWorld.ino décrit ci-dessous fait partie des quelques exemples proposés avec la bibliothèque Minitel1B.

Avec Minitel1B_Soft :

On commence par importer deux bibliothèques :

```
#include <Minitel1B_Soft.h>
#include <SoftwareSerial.h>
```

Puis on définit un port série logiciel pour la connexion avec le Minitel. J'utilise les broches 8 et 9 de la carte Arduino Uno pour la simple raison que j'ai mis au point un shield Arduino (3615) qui utilise ces deux broches (voir mon site Web). On peut bien évidemment en changer :

```
Minitel minitel(8, 9); // RX, TX
```

Avec Minitel1B_Hard :

On n'importe qu'une seule bibliothèque :

```
#include <Minitel1B_Hard.h>
```

Puis on choisit un port série matériel pour la connexion avec le Minitel. Sur l'ATmega 1284P avec lequel je travaille, je choisis le deuxième port série (Serial1 / RXD1 TXD1) :

```
Minitel minitel(Serial1);
```

Commun à Minitel1B_Soft et Minitel1B_Hard :

La connexion avec l'ordinateur se fait par le port série matériel Serial (RXD TXD pour l'ATmega 328P, RXD0 TXD0 pour l'ATmega 1284P). À la mise sous tension du Minitel, la vitesse des échanges entre le Minitel et le périphérique est de 1200 bauds par défaut. On envisage cependant le cas où le Minitel se trouve dans un autre état, ce qui peut arriver lorsque vous serez amené à bidouiller avec un Minitel à chaud.

```
void setup() {
  Serial.begin(9600); // 9600 bauds
  minitel.changeSpeed(minitel.searchSpeed());
  minitel.newScreen();
}

void loop() {
  minitel.print("Hello World ! ");
}
```

Les fonctions disponibles

Les fonctions sont réparties en 11 catégories. Elles sont décrites dans la bibliothèque elle-même, dans le fichier d'entête (.h). Je vais juste les lister, ce qui vous permettra d'avoir un aperçu des possibilités :

Vitesse de la liaison série (3 fonctions) :
int changeSpeed(int bauds), int currentSpeed(), int searchSpeed()

Séparateurs (2 fonctions) :
void newScreen(), void newXY(int x, int y)

Modes du standard Télétel (6 fonctions) :
void textMode(), void graphicMode(), byte pageMode(), byte scrollMode(), byte modeMixte(), byte modeVideotex()

Standards (2 fonctions) :
byte standardTeleinformatique(), byte standardTeletel()

Contenu (12 fonctions) :
void attributs(byte attribut), void print(String chaine), void println(String chaine), void println(), void printChar(char caractere), void printDiacriticChar(unsigned char caractere), void printSpecialChar(byte b), byte getCharByte(char caractere), void graphic(byte b), void graphic(byte b, int x, int y), void repeat(int n), void bip()

Curseur (10 fonctions) :
void cursor() - void noCursor(), void moveCursorXY(int x, int y), void moveCursorLeft(int n), void moveCursorRight(int n), void moveCursorDown(int n), void moveCursorUp(int n), void moveCursorReturn(int n), int getCursorX(), int getCursorY()

Effacements, Suppressions, Insertions (13 fonctions) :
void cancel(), void clearScreenFromCursor(), void clearScreenToCursor(), void clearScreen(), void clearLineFromCursor(), void clearLineToCursor(), void clearLine(), void deleteChars(int n), void insertChars(int n), void startInsert(), void stopInsert(), void deleteLines(int n), void insertLines(int n)

Géométrie (3 fonctions) :
void rect(int x1, int y1, int x2, int y2), void hLine(int x1, int y, int x2, int position), void vLine(int x, int y1, int y2, int position, int sens)

Clavier (6 fonctions) :
unsigned long getKeyCode(bool ascii = true), byte smallMode(), byte capitalMode(), byte extendedKeyboard(), byte standardKeyboard(), byte echo(boolean commande)

Protocole (4 fonctions) :
byte aiguillage(boolean commande, byte emetteur, byte recepteur), byte statusAiguillage(byte module), byte connexion(boolean commande), byte reset()

Écrire un octet ou un mot / Lire un octet (3 fonctions) :
void writeByte(byte b), void writeWord(word w), byte readByte()

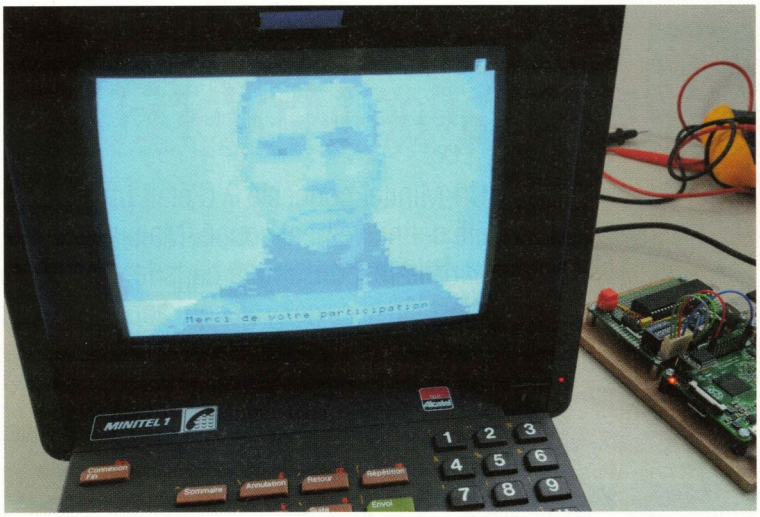
Parmi ces fonctions, l'une d'elle est vraiment particulière, c'est la fonction attributs dans la catégorie contenu. Elle permet plusieurs choses comme de fixer la couleur d'un caractère (CARACTERE_NOIR / CARACTERE_ROUGE / CARACTERE_VERT / CARACTERE_JAUNE CARACTERE_BLEU / CARACTERE_MAGENTA / CARACTERE_CYAN / CARACTERE_BLANC) ou sa couleur de fond (FOND_NOIR / FOND_ROUGE / FOND_VERT / FOND_JAUNE / FOND_BLEU / FOND_MAGENTA / FOND_CYAN / FOND_BLANC). Elle permet également de jouer sur la taille des caractères (GRANDEUR_NORMALE / DOUBLE_HAUTEUR / DOUBLE_LARGEUR / DOUBLE_GRANDEUR), de les faire clignoter ou pas (CLIGNOTEMENT / FIXE), d'inverser le fond ou pas (INVERSION_FOND / FOND_NORMAL), de les masquer (MASQUAGE / DEMASQUAGE). L'attribut de lignage (DEBUT_LIGNAGE / FIN_LIGNAGE) correspond à un soulignage en mode texte et à des caractères disjoints en mode semi-graphique. Pour utiliser la fonction attributs, il suffit de taper par exemple :

```
minitel.attributs(CARACTERE_NOIR);
minitel.attributs(FOND_JAUNE);
minitel.attributs(DOUBLE_HAUTEUR);
minitel.print("Hello World ! ");
```

Une autre fonction, qui sera beaucoup moins utilisée, fait également appel à des constantes particulières : il s'agit de printSpecialChar dans la catégorie contenue. Les possibilités sont : LIVRE / DOLLAR / DIESE / PARAGRAPHE / FLECHE_GAUCHE / FLECHE_HAUT / FLECHE_DROITE / FLECHE_BAS / DEGRE / PLUS_OU_MOINS / DIVISION / UN_QUART / UN_DEMI / TROIS_QUART / OE_MAJUSCULE / OE_MINUSCULE / BETA. Pour utiliser cette fonction, il suffit de taper par exemple :

```
minitel.printSpecialChar(FLECHE_HAUT) ;
```

Dans tous les cas, je vous invite à tester les quelques exemples proposés avec la bibliothèque, notamment Demo.ino pour ce qui concerne les possibilités d'affichage.



Afficher une image sur l'écran

Outre le texte, la bibliothèque permet également d'afficher des images. Pour cela la fonction clé est **writeByte** (écrire un octet). L'image, stockée sous forme d'instructions dans la mémoire Flash du microcontrôleur de la carte Arduino, nécessite en effet de lire une trame octet par octet et de l'aiguiller vers le Minitel. L'exemple Portrait.ino qui accompagne la bibliothèque donne un aperçu des possibilités en matière graphique. Obtenir ce type d'images à l'écran demande un peu de préparation. Je suis parti d'une image au format JPG, puis en utilisant le programme minitel11 qui fonctionne sous Processing 3 (https://drive.google.com/file/d/0BxdNVN-CFfF5aTdUUU1uREsxZ3M/view), j'ai pu créer un fichier Vidéotex correspondant à l'image en question, au format VDT. Reste ensuite à convertir ce fichier Vidéotex en suite d'octets pour le programme Arduino (https://github.com/eserandour/Conversion_Videotex_Hex). Le but de cet article n'étant pas l'affichage Vidéotex, je m'en tiendrai là, il s'agit juste d'ouvrir des perspectives.

Lorsque j'ai démarré ce projet de bibliothèque, j'avais également le désir de réaliser une sorte d'ordinateur 8 bits qui aurait fonctionné avec un Minitel. Je n'ai pas encore pris le temps de développer cette idée, cependant d'autres personnes se sont emparées de la bibliothèque Minitel1B_Hard et ont quelque part amorcé ce que je souhaitais faire. J'en suis très heureux. Il s'agit du projet ZARDOS de Lomig Perrotin (https://zardos.fr) et du projet Minitel ESP32 de Iodeo (https://github.com/iodeo/Minitel-ESP32). Merci à eux.

Quelques idées de projets

Un de mes premiers projets fut de réaliser une plate-forme dédiée à l'envoi de tweets depuis un Minitel (tutoriel sur mon site Web). J'ai choisi pour ce projet d'utiliser une carte Arduino Uno et un Raspberry Pi 2. Mais pourquoi donc utiliser une carte Arduino ? Je voulais que lorsqu'on allume le Minitel (qui alimente électriquement la carte Arduino), l'interface apparaisse tout de suite ; un microcontrôleur permet cette rapidité.