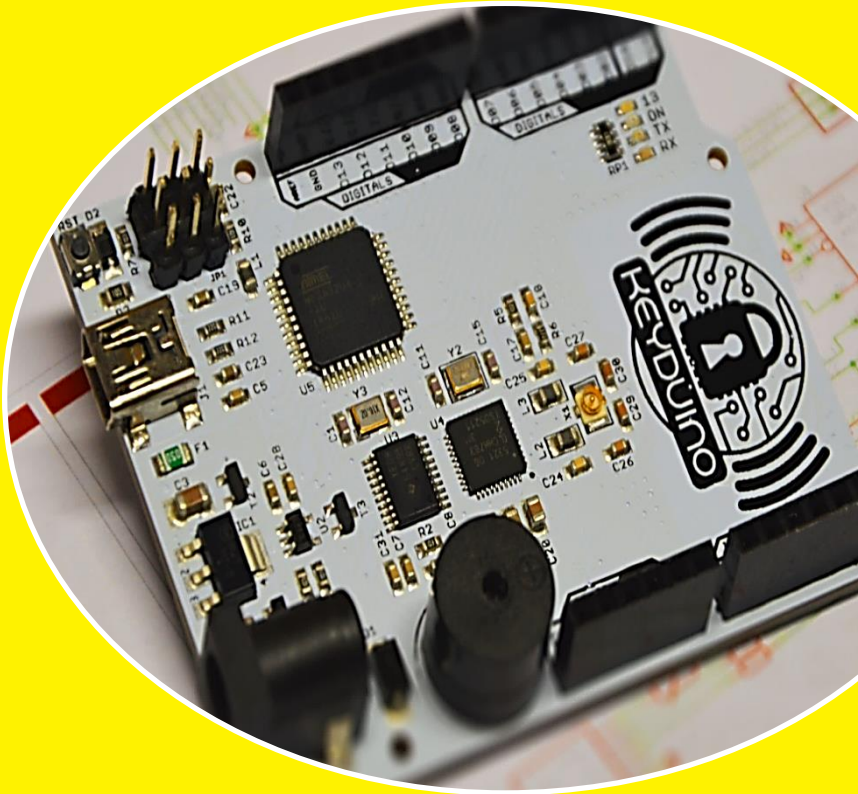
Making Everything Easier!™

KeyDuino

FOR DUMMIES®



Avertissements

KeyDuino est une plateforme de développement NFC qui permet de créer facilement et rapidement des prototypes. Cette carte ne consiste pas en un produit terminé pour telle ou telle application et nécessite en fonction du projet à réaliser d'y ajouter des équipements matériels ou logiciels à sécuriser

À ce titre, la question de la fiabilité et de la protection de vos développements à partir de la plateforme est importante et est de votre entière responsabilité, au même titre que l'usage des applications que vous développerez.

Par ce document, nous vous apportons quelques clefs nécessaires au bon fonctionnement de vos projets.

KeyDuino peut être un outil puissant, c'est pourquoi il nous revient collectivement de respecter, lors de son utilisation, les valeurs qui ont mené à sa création : la création d'une communauté de bidouilleurs/geeks/passionnés et surtout responsables s'attachant à développer des projets sympathiques, à partager les compétences et bonnes idées

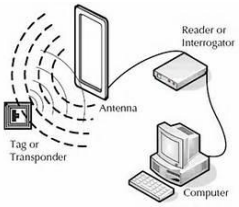
Dès lors, n'oubliez pas que les tentatives de hack sur des systèmes ne vous appartenant pas vont à l'encontre de « l'état d'esprit KeyDuino ». et que seul on va plus vite mais ensemble, on va plus loin !

KeyDuino Team

1) Introduction à la RFID

Présentation globale

La RFID (**R**adio **F**requency **I**dentification) est une technologie bien implantée dans de nombreux secteurs. Cette technologie existe depuis les années 1970, mais s'est surtout développée à la fin des années 1990.



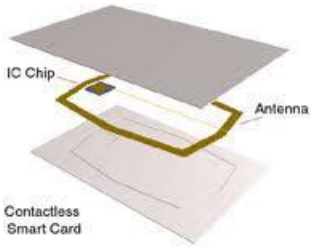
Les systèmes utilisant la RFID sont composés de deux parties : un lecteur, qui va venir apporter l'énergie, et un transpondeur (un tag).
Ces derniers communiquent par liaison radio fréquences.

On distingue 3 catégories d'applications :

- La RFID LF (Low Frequency) : de 125 à 135 kHz et ayant une portée de moins de 10 centimètres. Elle est principalement utilisée dans les badges d'accès.

- La RFID HF (High Frequency) : sur une fréquence de 13,56 MHz. La RFID HF est notamment utilisée dans les passeports biométriques et sert de base à la technologie NFC (Near Field Communication) présente dans de très nombreux produits.

- La RFID UHF (Ultra High Frequency): en Europe, sa fréquence est de 868 MHz. La portée est étendue à une dizaine de mètres. Cette technologie est notamment utilisée pour la traçabilité en logistique, ainsi que pour dresser des inventaires.



La structure d'un transpondeur est extrêmement simple. Il s'agit uniquement d'une antenne reliée à une puce.

Le lecteur se couple au transpondeur et envoie de l'énergie au transpondeur dont le processeur est alors alimenté.
Dans ce document, nous parlerons uniquement de tags passifs (non alimentés de façon interne).

En termes de couplage, il en existe deux types;

- le couplage **magnétique** (H) dans le cas d'un champ dit proche (de quelques centimètres à 1.5m). Les fréquences utilisées sont les LF et HF. Les antennes sont alors constituées de boucles inductives.
- le couplage **électrique** (E) pour le champ dit lointain (jusqu'à 6 mètres).
  Les fréquences en jeux sont l'UHF. Les antennes sont alors des dipôles ou des patchs.

La NFC

La NFC, ou Near Field Communication (Communication en Champ Proche), est une technologie standardisée de communication sans-fil à courte portée, permettant des transactions, de l'échange de données numériques, et la connexion d'appareils électroniques par un simple contact. La NFC est compatible avec des centaines de millions de smartphones, cartes sans-contact, et lecteurs déjà déployés dans le monde entier.

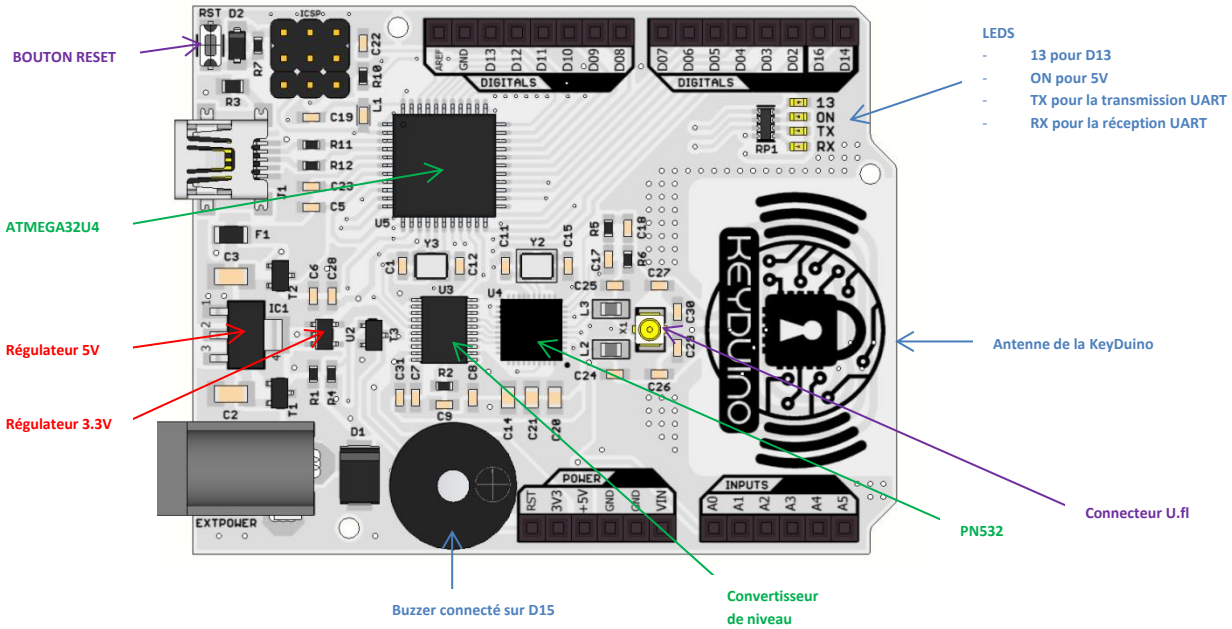
2) Introduction à KeyDuino

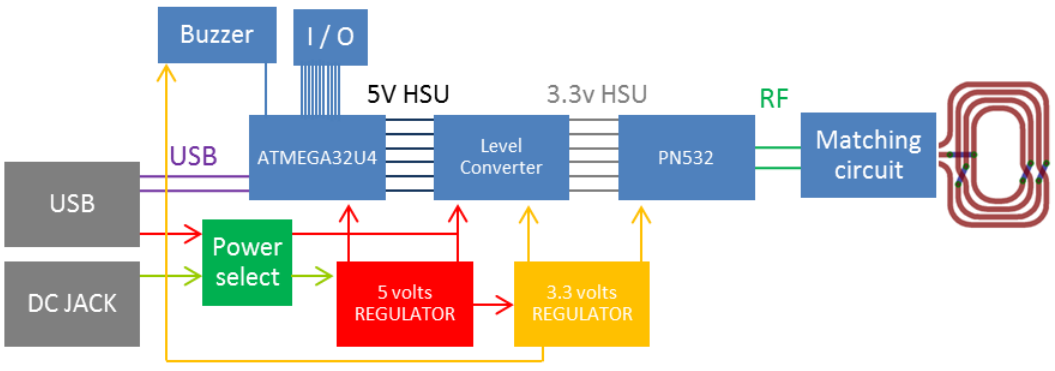
Généralités sur KeyDuino

KeyDuino est une carte électronique programmable utilisant l'environnement Arduino (une plateforme de développement open source) et permettant de créer des interactions avec la technologie NFC.
Il est possible d'imaginer de nombreux scénarios d'utilisation des cartes KeyDuino dans de multiples domaines (transfert d'informations par NFC, identification sur une chaine de production,…). :

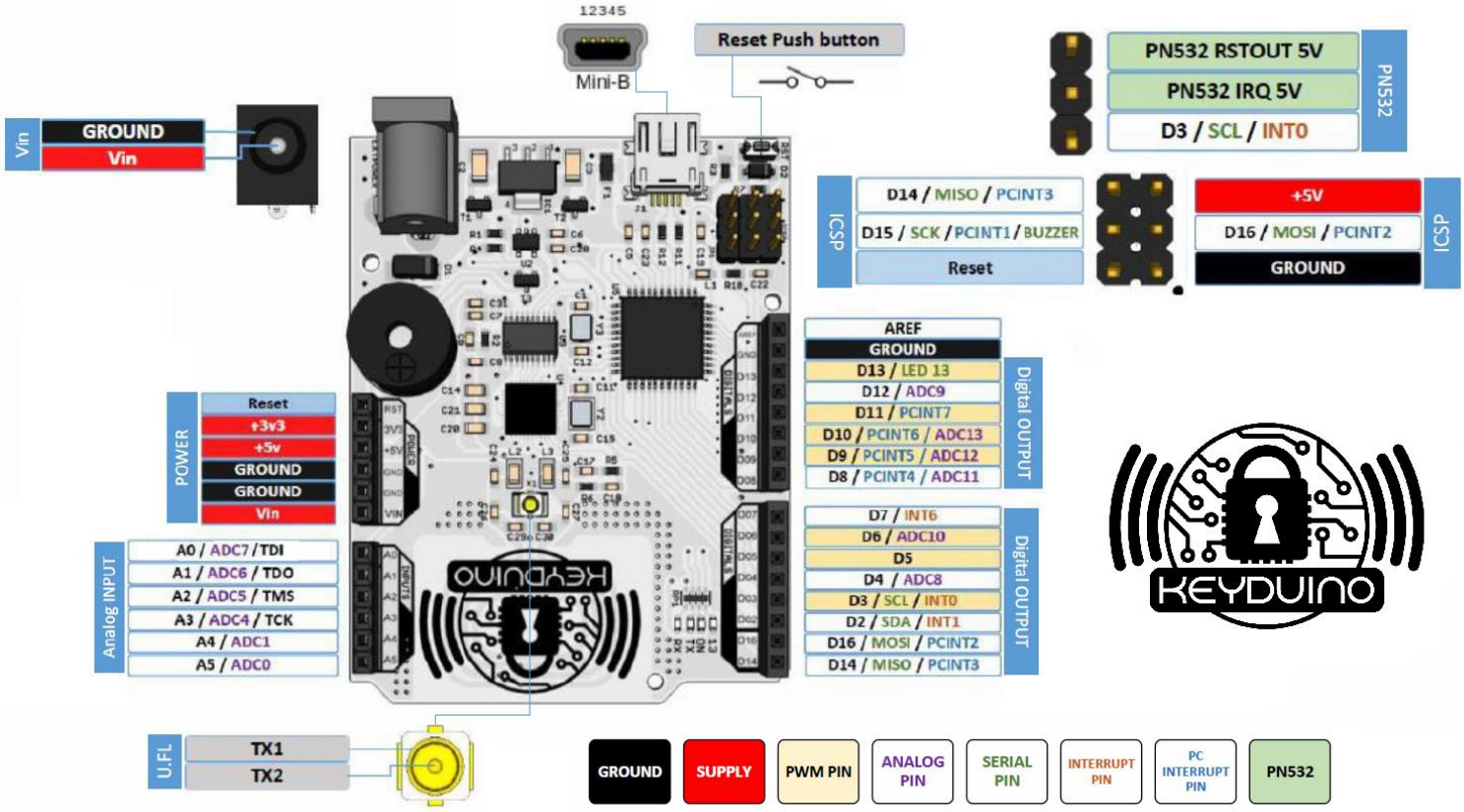
L'objectif principal de KeyDuino est de proposer une solution de prototypage rapide, plus spécifiquement dans le domaine de la sécurité. En effet, grâce aux nombreuses entrées sorties (E/S) dont disposent les cartes, connecter un actionneur, mécanique par exemple, est une chose facile. Dès lors, interagir sur une gâche de fermeture de porte, un verrou électronique, un starter de voiture, (…), est possible.
La technologie NFC prend tout son intérêt dans ce type d'applications. La lecture en champs proche permet d'opter pour une identification par carte, par badge et surtout via une bague NFC ou un smartphone.

Architecture globale de KeyDuino





Pinout de KeyDuino



Alimentations

- Alimentation 5v
- Alimentation 3.3v
- Entrée d'alimentation VIN (pin ou connecteur)
- Sélection automatique entre VIN et l'alimentation 5volts

Entrée/sorties

- 21 entrées sorties utilisables (A0-A5 & D2/D15)
- 7 PWM (D3, D5, D6, D9, D10, D11, D13)
- 12 entrées (A0-A5 & D4, D6, D8, D9, D10, D12)

Communications

- Software UART
- Software SPI
- I2C

Processeur

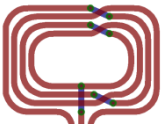
- ATMEGA32U4
- mémoire flash disponible : 28KB
- SRAM : 2.5KB
- EEPROM : 1KB
- Fréquence 16Mhz

NFC

- PN532 connecté en HSU
- IRQ accessible
- RSTOUT accessible
- Connecteur U.fl lié à TX1, TX2

Fréquence en jeu

Connaitre la fréquence de fonctionnement de KeyDuino va vous permettre de comprendre quel type de tags la carte pourra lire et ceux avec lesquelsi il sera impossible de communiquer.

Pour ce faire, observons un peu la forme de l'antenne de la KeyDuino : 

Comme vous pouvez le constater, il s'agit d'une bobine constituée d'un ensemble de boucles inductives (4 boucles). KeyDuino utilise donc (principalement) le champ magnétique E pour communiquer avec les tags en champ proche.
Pour préciser, l'antenne de KeyDuino et son circuit d'accord sont calibrés pour une fréquence de 13.56Mhz, soit la fréquence de la RFID HF (excluant de fait la lecture de tags LF ou UHF).

125 KHz

13.56 MHz

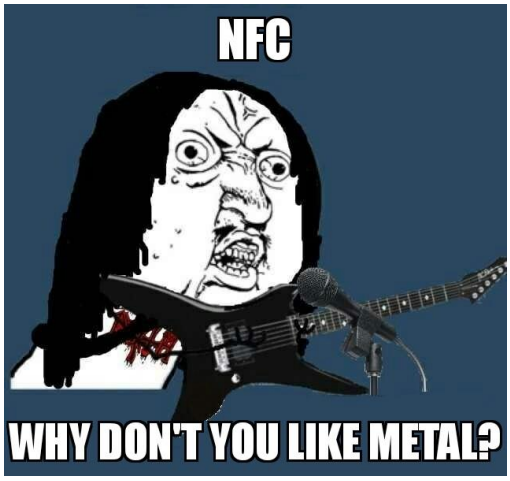
868 MHz

Le tranceiver utilisé pour permettre la lecture des tags est un PN532 connecté en HSU (High Speed Uart). Ce module supporte 7 modes d'opérations différents :

- ISO/IEC 14443A/MIFARE Reader/Writer
- FeliCa Reader/Writer
- ISO/IEC 14443B Reader/Writer
- ISO/IEC14443A/MIFARE Card MIFARE Classic 1K
- MIFARE Classic 4K card emulation mode
- FeliCa Card emulation
- ISO/IEC 18092, ECMA 340 Peer-to-Peer

Métal et KeyDuino

Le champ magnétique de la NFC ne peut fonctionner à travers le métal. Par conséquent, il est impossible de lire un tag à travers n'importe quel élément métallique via la KeyDuino ou une antenne déportée.



Détruire correctement votre KeyDuino

KeyDuino vous permettra de prototyper bon nombre de projets. Néanmoins, dans la précipitation, certaines erreurs classiques peuvent endommager, voire détruire votre KeyDuino.
Voici une liste non-exhaustive des erreurs potentielles :

Court circuits :

- Faire un court-circuit entre une entrée/sortie et la masse/l'alimentation
- Faire un court-circuit entre deux entrées/sorties avec chacune un état différent
- Faire un court-circuit entre l'alimentation 5v/3.3v/VIN et la masse

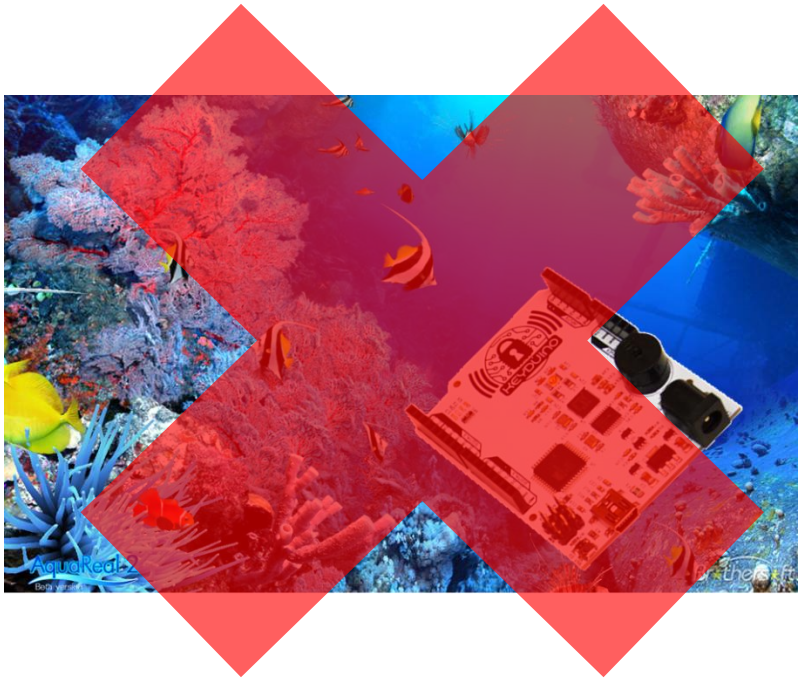
Surtension :

- Appliquer une tension supérieure à 5.5volts sur une entrée/sortie
- Appliquer une tension supérieure à 5 volts sur la pin 5volts
- Appliquer une tension supérieure à 12 volts sur la pin 12 volts / le connecteur DC

Surconsommation :

- Consommer plus de 40mA par pin
- Consommer plus de 200mA sur l'ensemble des pins
- Consommer plus de 2A sur l'alimentation VIN
- Consommer plus de 500mA sur l'alimentation 5volts
- Consommer plus de 150mA sur l'alimentation 3.3volts

Etonnamment, KeyDuino n'est pas prévu pour fonctionner à des températures extrêmes, des milieux trop humides ou sous des chocs trop importants. Des erreurs classiques telles que mesurer les vibrations à l'intérieur d'un mixer ou la température dans un four peuvent laisser quelques dommages à KeyDuino.



Prenez donc soin de votre KeyDuino !

3) Initialiser votre KeyDuino

Installer l'environnement Arduino

Pour commencer à utiliser et programmer votre KeyDuino, vous aurez besoin de l'environnement de développement (IDE) Arduino. Vous pouvez le télécharger ici :

https://www.arduino.cc/en/Main/Software

Les codes actuels sont développés avec la version 1.6.5 d'Arduino. Si vous débutez avec Arduino, il est recommandé d'utiliser la même version pour éviter tous problèmes.
Décompressez le fichier et installez Arduino.

Télécharger la bibliothèque KeyDuino

Pour avoir accès aux codes d'utilisations et à la bibliothèque KeyDuino, il faut premièrement la télécharger sur la plateforme GitHub :

https://github.com/CITCEuraRFID/KeyDuino

Cliquer sur le bouton « Download ZIP », et une fois téléchargé, l'extraire. Dans le répertoire KeyDuino, vous trouverez divers dossiers, comme hardware, qui contient les schémas électroniques notamment. Copiez simplement le sous-dossier KeyDuino, et collez le dans le répertoire « libraries » d'Arduino.
La bibliothèque est alors installée ! Il n'y a plus qu'à lancer l'IDE Arduino pour pouvoir l'utiliser.

Télécharger l'application KeyDuino

Pour utiliser l'application Android développée spécialement pour KeyDuino, télécharger l'apk sur la plateforme GitHub.
Vous pouvez également trouver les sources de cette application dans le dossier de la bibliothèque KeyDuino précédemment téléchargé.
Signalons que bien malheureusement, les Apple iPhones ne sont pas compatibles avec KeyDuino ou les tags NFC dans leurs versions actuelles.

Installer les drivers KeyDuino

Connectez la KeyDuino à un port USB avec le câble fourni.
La LED « On » doit s'allumer.

- **Windows**

Pour Microsoft Windows XP / Seven :

http://www.visualmicro.com/post/2012/06/02/Arduino-Leonardo-Windows-Hardware-USB-Installation-Guide.aspx

- **Linux**

Connectez la KeyDuino. C'est tout.

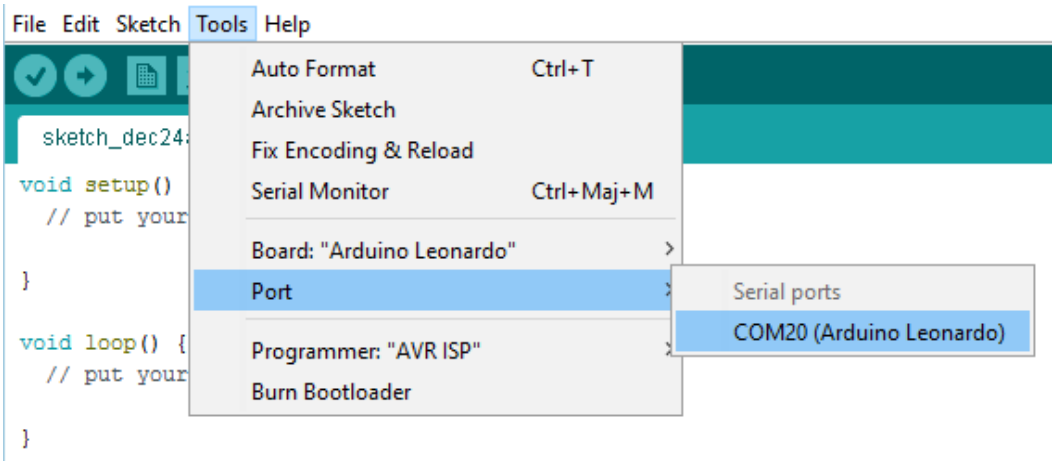
- **Mac OS**

La première fois que vous connectez KeyDuino sur un système Mac OS, le "Keyboard Setup Assistant" s'ouvrira. Il n'y a rien à configurer, il vous suffit juste de fermer la fenêtre.

4) Débuter avec votre KeyDuino

Identifier un tag

Sélectionnez votre port sur lequel est connectée votre KeyDuino. Comme la carte possède le même bootloader qu'un Arduino Leonardo, il devrait être identifié comme un Arduino Leonardo.



Erreur : Vous ne voyez pas le port

Solution 1 : Déconnectez votre KeyDuino, fermez l'IDE Arduino. Reconnectez KeyDuino et retentez de sélectionner votre port
Solution 2 : Réinstallez les drivers, assurez-vous que KeyDuino est bien reconnu par votre ordinateur

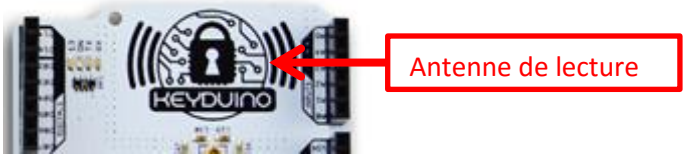
Normalement, le sketch de base permet de lire le tag NFC fourni et d'afficher les ID obtenus.
Pour voir les informations, cliquer sur « Serial Monitor » en haut à droite de la fenêtre Arduino.

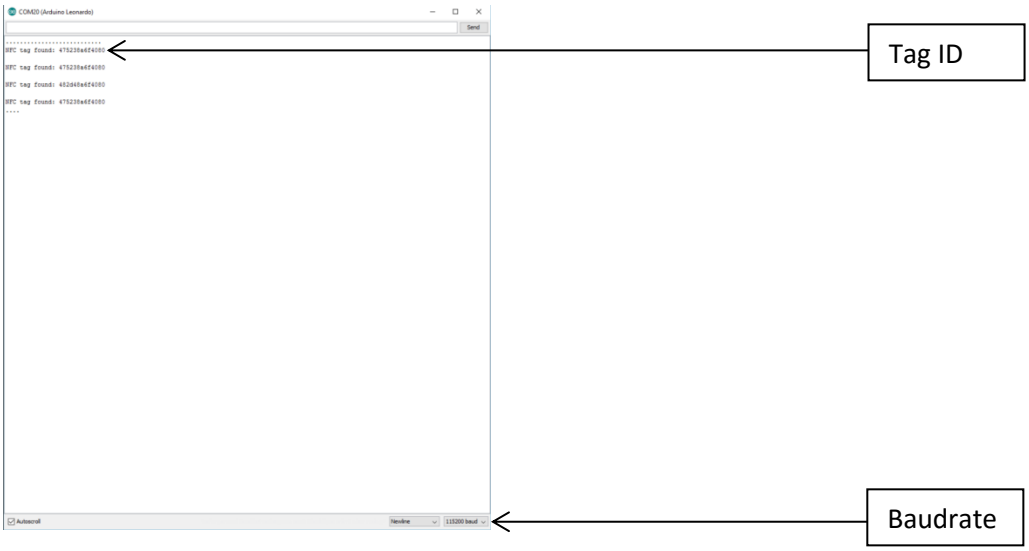
Erreur : vous rencontrez le message d'erreur suivant

Vérifiez si le port com de la KeyDuino est toujours le même, il peut arriver que le port se déconnecte/reconnecte (principalement pendant le téléversement d'un programme. Puis tentez de rouvrir à nouveau le moniteur série.
Si le problème persiste, débranchez la KeyDuino, fermez le moniteur rebranchez la KeyDuino et retentez de vous connecter.

Pour cet exemple, votre baudrate doit être à 115200Baud, par défaut, si vous venez de télécharger Arduino, il devrait être à 9600 : Ill faudra donc le changer en bas à droite de Serial Monitor.

Passez à présent l'un des tags fourni prêt de l'antenne de lecture de la KeyDuino, normalement, a plus de 1 centimètre, le buzzer bipe un coup et l'id du tag s'affiche dans la console du moniteur série.





Erreur : Rien ne se passe

Solution 1 : Essayez avec le second tag, si celui-ci réagit, merci de nous contacter pour vous remplacer le tag défectueux.

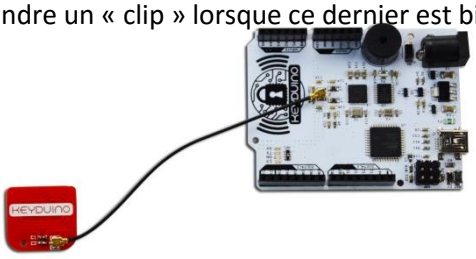
Solution 2 : Eloignez la KeyDuino de toute zone métallique lors de la lecture.

Solution 3 : Testez l'exemple PN532_TEST ; si KeyDuino ne bipe pas, merci de nous contacter (keyduino@outlook.com).

Si la KeyDuino bipe, essayez de programmer la KeyDuino avec le programme tag_identification (voir plus bas).

(Optionnel) Connection de votre antenne déportée

Si vous avez commandé une antenne déportée avec votre KeyDuino, connectez votre câble selon le schéma joint, vous devez entendre un « clip » lorsque ce dernier est bien enfoncé. C'est tout.



Conseils :

- Essayez de ne pas tordre le câble
- Ne connectez/déconnectez pas le câble trop souvent, le connecteur est fragile et doit être manipulé avec soin
- Évitez de le passer entre des endroits métalliques
- Comme pour l'antenne intégrée de la KeyDuino, ne collez pas l'antenne déportée à du métal
    - Utilisez des vis plastiques plutôt que métalliques pour fixer l'antenne.
    - Sachez que la super glue / le scotch double face marche très bien !

Vous êtes maintenant prêt à utiliser pleinement votre KeyDuino !

5) Utiliser votre KeyDuino

Voici maintenant quelques exemples classiques d'utilisation basique de KeyDuino.

Récupérer l'identifiant (UID) d'un tag

Lors de la manipulation de tags RFID, la première action côté lecteur est généralement de lire l'identifiant unique du tag. L'exemple ci-dessous devrait être compatible avec la plupart des types de tags RFID HF que vous pourriez trouver.

Dans l'IDE Arduino, lancer l'exemple tag_identification :

File > Examples > KeyDuino > tag_identification

Dans cet exemple, la fonction suivante :

```
keyDuino.readTargetID(uid, &uidLength)
```

récupère l'UID du tag présenté à l'antenne du KeyDuino, et stocke ce résultat dans la variable `uid` (et sa taille dans la variable `uidLength`).

Attention !

Les trois exemples suivant nécessitent l'installation de la bibliothèque NDEF, que vous pouvez trouver à l'adresse suivante :

https://github.com/don/NDEF

NDEF, pour NFC Data Exchange Format, est un protocole standard de définition des messages transmis via NFC ; un tag avec du contenu NDEF, une fois scanné par un smartphone par exemple, déclenchera sur l'appareil le lancement d'une URL, composera un numéro de téléphone, etc.

Lire le contenu d'un tag NDEF

Pour lire les données contenues dans un tag formaté NDEF, lancer l'exemple suivant :

File > Examples > KeyDuino > NDEF > read_tag

Une fois le code envoyé à la carte KeyDuino, ouvrez le Serial Monitor, et passez un tag sur l'antenne de la carte ; vous devriez alors voir s'afficher les données qu'il contient.

Écrire en NDEF dans un tag

Lancez l'exemple d'écriture dans un tag formaté NDEF :

File > Examples > KeyDuino > NDEF > write_tag

Choisissez un contenu à écrire dans le tag : URL d'un site web, numéro de telephone, adresse mail … Pour que le contenu soit fonctionnel, il faut qu'il soit correctement formaté. Vous pourrez trouver sur Internet comment fonctionne le protocole NDEF.

Vous pouvez ensuite téléverser le programme sur votre KeyDuino, et passer un tag sur l'antenne pour écrire dessus. Testez ensuite avec votre KeyDuino (exemple read_tag) ou un smartphone NFC pour vérifier que le contenu a bien été écrit sur le tag.

Peer to Peer

Pour vous permettre de prendre en main la communication Peer to Peer (P2P) en NFC, nous avons développé un sketch d'exemple pour la carte, ainsi qu'une application mobile Android. Via un smartphone compatible, vous pourrez ainsi communiquer et interagir avec votre KeyDuino depuis l'application : contrôler des pins, lire les valeurs de tension en sortie …

La démarche à suivre pour vous procurer l'application Android est décrite en Page 8 de ce document. Le sketch d'exemple se trouve à l'emplacement suivant :

File > Examples > KeyDuino > keyduino_android_app

6) Utilisez une shield compatible Arduino

Software

La carte est totalement compatible d'un point de vue software avec une carte Arduino Leonardo. Dès lors, n'importe quel code fonctionnant sur Leonardo fonctionnera sur KeyDuino.

UART

Pour faire la connexion avec le PN532 en HSU, nous avons utilisé TX (D1) & RX (D0). Néanmoins comme de nombreuses shield Arduino utilisent l'UART, nous avons déporté D14 & D16 de ICSP pour pouvoir les utiliser en Software UART.

Pour utiliser SoftwareSerial, suivez ce lien :

http://www.arduino.cn/reference/SoftwareSerial.html

Dans notre cas, la définition de RX et TX se fera de la façon suivante :

```
SoftwareSerial mySerial(14, 16);
```

SPI

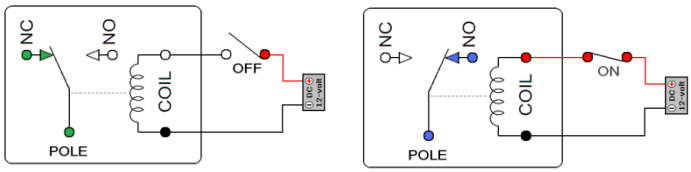
Pour pouvoir placer l'antenne, nous avons repositionné le connecteur ICSP utilisé pour le SPI sur une carte Arduino Leonardo. Pour pouvoir l'utiliser ; il y a deux choix :

- Utiliser des connecteurs entre le ICSP de la shield et celui de KeyDuino (vous ne serez plus en mesure d'utiliser le software UART et le buzzer !).
- Redéfinir le SPI sur d'autre PIN via SoftwareSPI (ex : 13, 12, 11) et effectuer les jonctions avec des connecteurs si c'est nécessaire (quelques shields comme la shield SD n'ont pas de connecteur ICSP).

7) (Optionnel) Utiliser la shield relais

Sans alimentation à ses bornes, un relais est bloqué dans sa position Normaly Closed (NC). Dans ce cas, il y a une jonction physique entre le COM et le NC.

En mettant une tension à ses bornes (5V dans notre cas), le courant excite une bobine qui vient commuter le contact. Le contact se fait alors entre COM et Normally Open (NO).



Dans notre cas, l'interrupteur ON/OFF correspond à la commande de la pin de la KeyDuino. C'est à dire que si la pin est à l'état haut, il y aura jonction entre NO et COM.

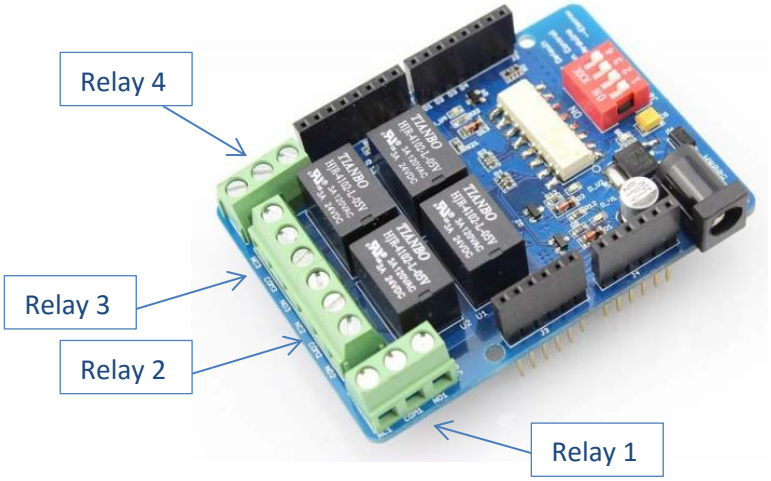
Dans la mesure où la shield relais possède 4 relais, cette dernière sera commandée par 4 pins :

Relay1 = KeyDuino pin D7

Relay2 = KeyDuino pin D6

Relay3 = KeyDuino pin D5

Relay4 = KeyDuino pin D4



Vous pouvez trouver un sketch d'utilisation dans les exemples de la bibliothèque KeyDuino :

File > Examples > KeyDuino > relay_shield

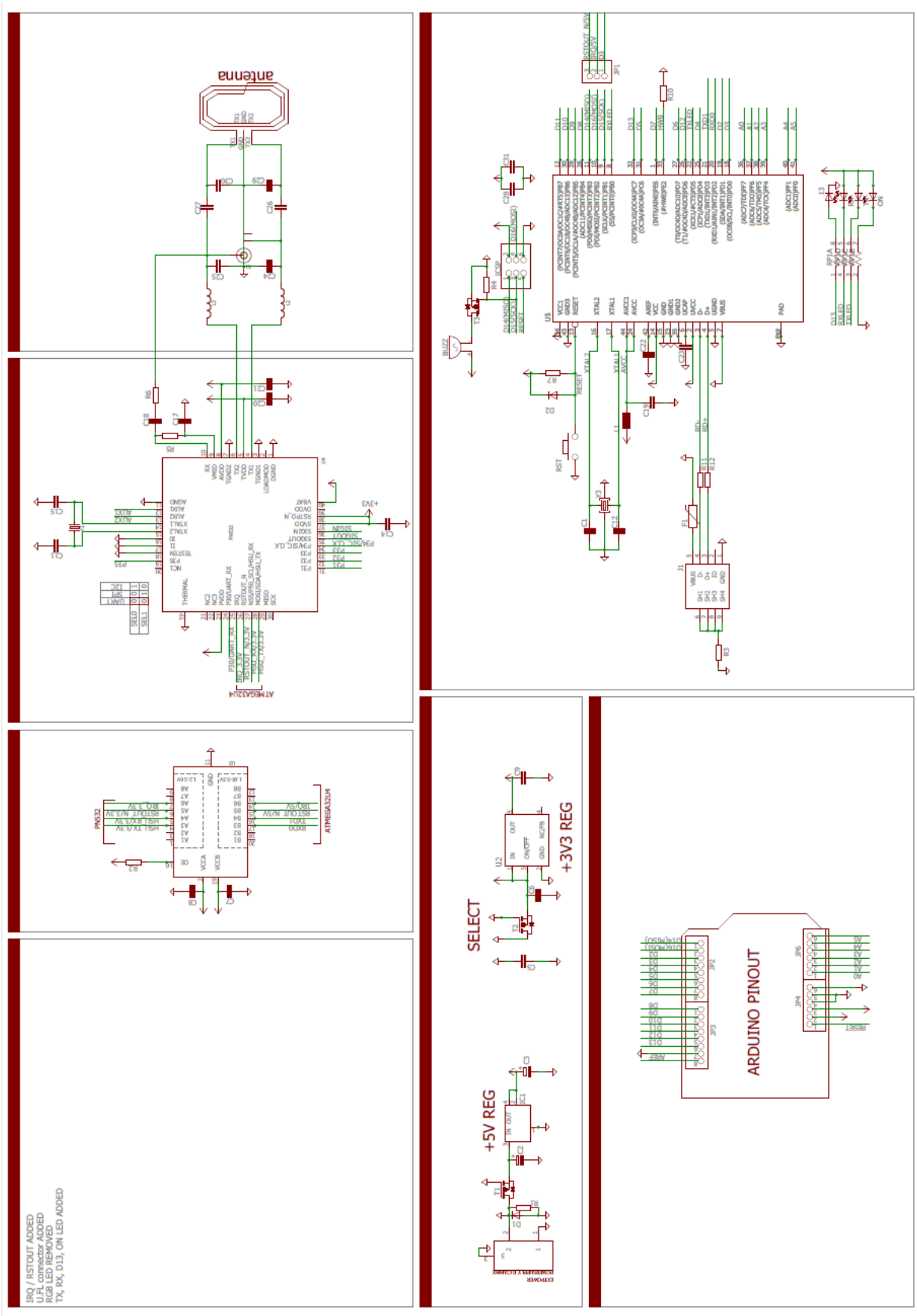
Cet exemple permet de fermer l'ensemble des contacts des relais sur l'identification d'un badge (selon un UID spécifique, ou quel que soit l'UID). Vous pouvez donc compiler et téléverser votre programme dans la KeyDuino en ayant sélectionné « Arduino Leonardo » et le port Com utilisé.

Si vous souhaitez que les relais se ferment lors de la détection d'un UID précis, renseignez le chant **RIGHT_UID[4],** par défaut valant « **FF FF FF FF** » avec l'UID spécifique souhaité.

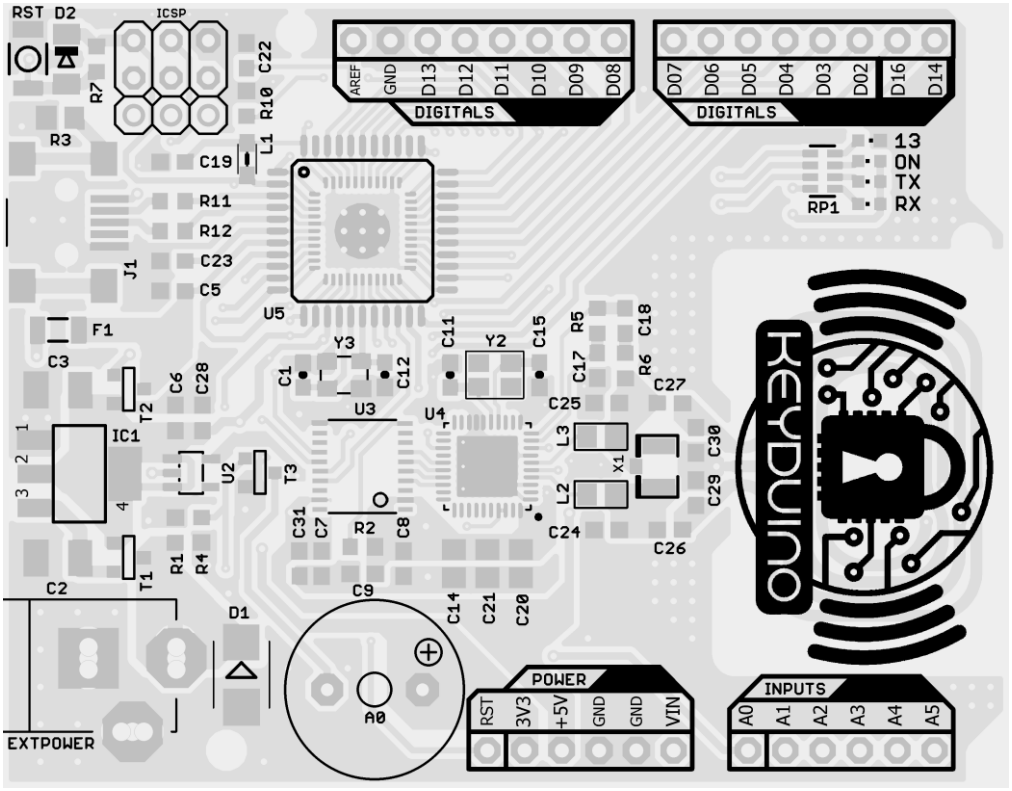
Pour plus d'informations, consultez le wiki d'ELECROW sur cette shield :

http://www.elecrow.com/wiki/index.php?title=Relay_Shield

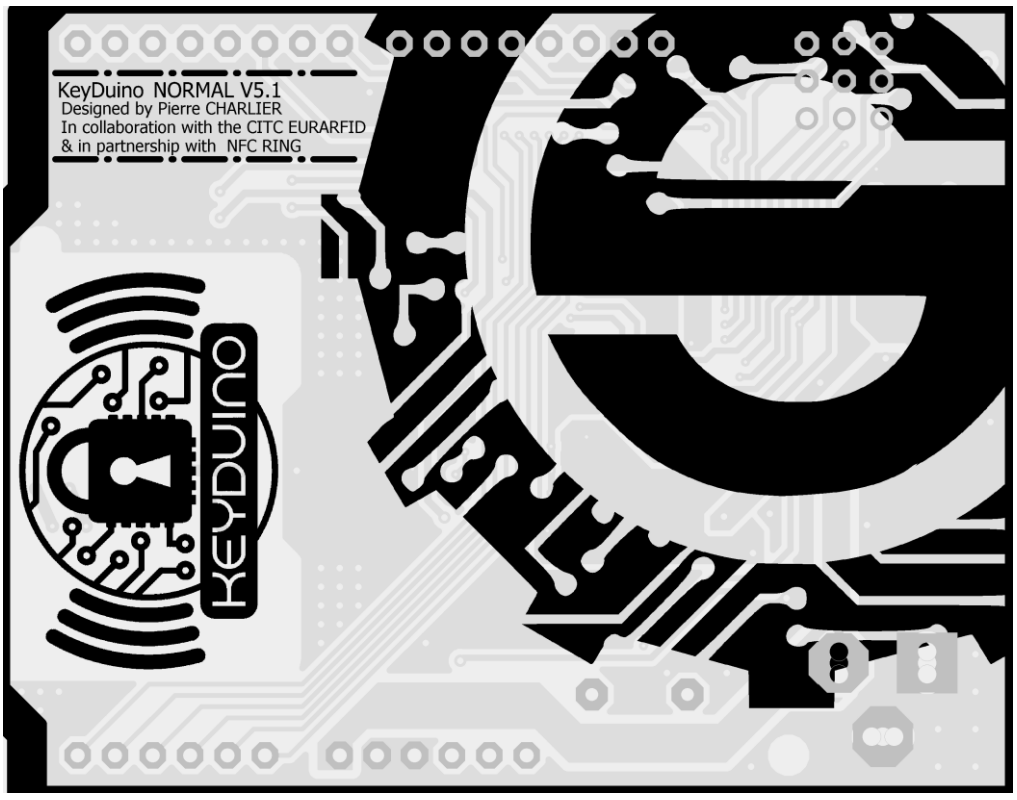
Annexe 1 : Schéma électronique



Annexe 2 : Layout



TOP



BOTTOM

Annexe 3 : BOM (Bill Of Material)

Part	QTY	Value	Device	Package	Description	REF
CAPACITOR						
C1, C11, C12, C15	4	22pF	C-EUC0603	C0603	Capacitor	capacitor 0603 22pF
C24, C25	2	220pF	C-EUC0603	C0603	Capacitor	capacitor 0603 220pF
C26, C27	2	2200pF	C-EUC0603	C0603	Capacitor	capacitor 0603 2200pF
C18	1	1nF	C-EUC0603	C0603	Capacitor	capacitor 0603 1nF
C29, C30	2	1,2nF	C-EUC0603	C0603	Capacitor	Capacitor 0603 1,2nF
C5, C7, C8, C17, C22, C28, C31	7	100nF	C-EUC0603	C0603	Capacitor	capacitor 0603 100nF
C6, C9, C19, C23	4	1uF	C-EUC0603	C0603	Capacitor	capacitor 0603 1uF
C21	1	100nF	C-EUC0805	C0805	Capacitor	capacitor 0805 100nF
C14, C20	2	10uF	C-EUC0805	C0805	Capacitor	capacitor 0805 10uF
C2	1	10uF	CPOL-EUSMCB	SMC_B 1206	CAPACITOR	SMC_B 1206 10uF
C3	1	22uF	CPOL-EUSMCB	SMC_B 1206	CAPACITOR	SMC_B 1206 22uF
RESISTOR						
R11, R12	2	22	R-EU_R0603	R0603	RESISTOR	RESISTOR 0603 22
R5	1	1.0K	R-EU_R0603	R0603	RESISTOR	RESISTOR 0603 1K
R6	1	2.7K	R-EU_R0603	R0603	RESISTOR	RESISTOR 0603 2,7K
R1, R2, R4, R7, R10	5	10K	R-EU_R0603	R0603	RESISTOR	RESISTOR 0603 10K
R3	1	1M	R-EU_R0805	R0805	RESISTOR	RESISTOR 0805 1M
RP1	1	330	RES4NT	RES4NT	4 Resistor Array	4 Resistor Array 330
INDUCTOR						
L1	1	MH2029-300Y	WE-CBF_0805	805	SMD EMI Ferrite	MH2029-300Y
L2, L3	2	560nH	INDUCTOR0805	0805@1	Inductors	inductor 560nH 0805
DIODE & FUSE						
D1	1	M7	DIODE-SMB	SMB	DIODE	SMB M7 (leonardo)
D2	1	CD1206-S01575	DIODE-MINIMELF	MINIMELF	DIODE	CD1206-S01575
F1	1	500mA	PTCSMD	PTC-1206	Resettable Fuse PTC	FUSE 0805 500mA
CONNECTOR & MECHANICAL						
ICSP	1	ICSP	PINHD-2X3	2X03	PIN HEADER	
JP1	1		PINHD-1X3	1X03	PIN HEADER	
JP2	1		PINHD-1X8CLEANBIG	1X08-CLEANBIG	PIN HEADER	
JP3	1		PINHD-1X8CLEANBIG	1X08-CLEANBIG	PIN HEADER	
JP4	1		PINHD-1X6CB	1X06-CLEANBIG	PIN HEADER	
JP6	1		PINHD-1X6CB	1X06-CLEANBIG	PIN HEADER	
X1	1		ANTENNA_U.FL	U.FL	U.FL Antenna Connector	u.FL connector
J1	1	USB-MINI-B%C	USB-MINI-B%C	USB-MINI-B_2		
EXTPOWER	1	POWERSUPPLY_DC21MMX	POWERSUPPLY_DC21MMX	DC-21MM		
RST	1	RESET	PB157	157SW	SKRKAEE010	157SW
BUZZ	1	F/QMX	F/QMX	F/QMX		F/QMX (3,3v)
LEDs						
RX	1		LEDCHIPLED_0603	CHIPLED_0603	LED	0603 LED yellow
TX	1		LEDCHIPLED_0603	CHIPLED_0603	LED	0603 LED RED
ON	1		LEDCHIPLED_0603	CHIPLED_0603	LED	603 LED green
13	1		LEDCHIPLED_0603	CHIPLED_0603	LED	603 LED blue
IC						
U1	1	ATMEGA32U4-XUAU	ATMEGA32U4-XUAU	TQFP44-PAD	ATMEGA32U4-XUAU	ATMEGA32U4-XUAU
U2	1	LP2985-33DBVR	LP2985-XXDBVR33	SOT23-DBV	ULTRALOW-POWER REGULATORS	LP2985-33DBVR
IC1	1	NCP1117ST50T3G	MC33269ST-3.3T3	SOT223	Regulator 800 mA	NCP1117ST50T3G
U3	1		TXB0108PWR	TSSOP20	8-Bit Bi-Directional Level Shifter	TXB0108PWR
U4	1	PN532	PN532	HVQFN40-6X6	PN532 – NFC controller	PN532
CRYSTAL						
Y2	1	27.12MHz	CRYSTALTHIN	CRYSTAL_3.2X2.5	Crystals	7B-27,1200MAAJ-T
Y3	1	16MHz KX-7	CRYSTAL-3.2-2.5	CRYSTAL-3.2-2.5		16MHz KX-7
MOSFET						
T1	1	PMV48XP	PMOSSOT23	SOT-23	MOS FET	PMV48XP
T2	1	FDN340P	PMOSSOT23	SOT-23	MOS FET	FDN340P
T3	1	BSS123	PMOSSOT23	SOT-23	MOS FET	BSS123