

ARDUINO

LES CARTES ARDUINO ET LEURS UTILISATIONS	3
1 Introduction	3
2 Historique du projet Arduino.....	4
3 Documentation sur Arduino	5
4 Cartes Arduino Uno.....	6
4.1 Les différentes versions.....	6
4.2 Implantation et schéma Arduino uno Rev2.....	6
4.3 Alimentation.....	8
4.3.1 Schéma.....	8
4.3.2 Alimentation uniquement par la prise usb	8
5 E/S disponibles sur connecteurs de la carte Arduino Uno	9
6 Environnements de développement Intégrés (EDI) utilisables	10
6.1 EDI Arduino spécifique à la carte cible	10
6.1.1 Généralités.....	10
6.1.2 Rappels sur le langage / Fonctions disponibles	11
6.1.3 Compilation.....	12
6.2 EDI Scratch for Arduino spécifique à la carte cible.....	12
6.3 EDIs non spécifiques à la carte cible.....	13
6.3.1 Eclipse	14
6.3.2 Visual Micro = Arduino for Visual Studio (Microsoft)	14
6.3.3 AVR studio 5.....	14
6.4 Flowcode	14
6.5 LabView	14
7 Programmation du µC principal puis utilisation.....	15
7.1 Généralités sur la programmation et l'utilisation	15
7.2 Programmation ICSP (In Circuit Serial Programming)	15
7.3 Programmation via la liaison USB	15
7.3.1 Généralités.....	15
7.3.2 Principe	15
7.3.3 Installation du pilote USB.....	16
7.3.4 Carte Arduino vue du logiciel de programmation.....	17
7.3.5 Programmation depuis FlowCode.....	18
7.3.6 programmation avec AVRdude	18
7.3.6.1 Téléchargement d'AVRdude.....	18
7.3.6.2 Ligne de commande AVRdude.....	19
8 Cartes d'extension (shields).....	19
8.1 Cartes avec zone de câblage (+ borniers).....	20
8.2 Carte afficheur LCD liaison // LCD_KEYPAD	21
8.3 Afficheur LCD liaison I2C.....	22
8.4 Interface 2 moteurs Ardumoto	23
8.5 MotoProto Shield Modkit.....	24
8.5.1 Broches utilisées	24
9 Dimensions de la carte, emplacement des connecteurs.....	25
9.1 Carte cotée	25
9.2 Fichiers solidworks	25
9.3 Modèles OrCAD pour créer des cartes d'extension	25


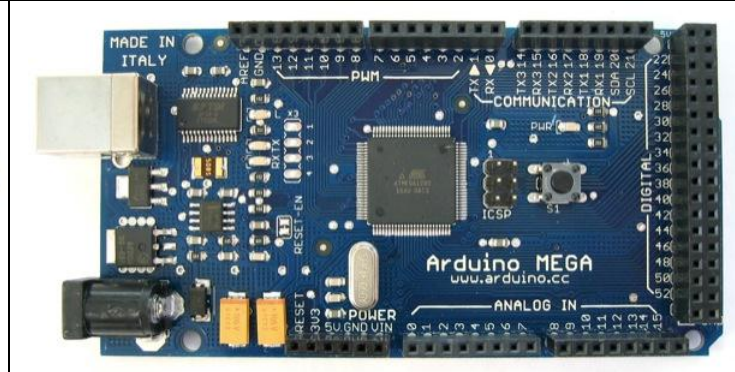
10 Carte d'interconnexion PEM	26
10.1 Généralités	26
10.2 Signaux Générés par la carte arduino avec flowcode	27
10.3 Schéma complet	28
10.4 Notes de conception	32
10.4.1 CAN	32
10.4.2 Alimentations	32
10.4.3 Tension délivrée par la batterie et seuils	32
10.4.4 Connecteurs	33
11 Carte LCD PEM	33
11.1 Connexions et compatibilités	33
11.2 Schéma structurel	34
11.3 Tension selon appui BP	34
11.4 Utilisation avec l'EDI Arduino	35
11.5 Utilisation avec Flowcode	36
12 Schématisation et fabrication de carte d'extension avec Fritzing	37
13 Quelques exemples d'utilisation d'Arduino	37
13.1 Robotique	37
13.1.1 Robots NXT de Lego	37
13.1.2 Boe Shield Bot	37
13.1.3 Pololu	38
13.2 Projets disponibles sur Internet	39
EDI ARDUINO	40
1 Langage de programmation	40
2 Arduino et Linux	40
3 Fonctionnement de la compilation	40
PROCESSING	42

LES CARTES ARDUINO ET LEURS UTILISATIONS

1 INTRODUCTION

Ce document est principalement consacré à l'utilisation d'Arduino dans un cadre scolaire, principalement dans la filière STI2D SIN (Sciences et Techniques Industrielles et Développement Durable Systèmes d'Information et Numérique). Les informations données ici permettent cependant de l'utiliser dans un autre cadre.

Les platines Arduino sont des cartes d'usage général à μC ATmega. Le fabricant les présente comme des platines de prototypage rapide. Il existe plusieurs familles de cartes. Voici 2 familles par exemple :

Famille de cartes autour de μC 28 broches, avec des μC ATmega 8 ou 168 ou 328P selon les modèles. 3 ports sont disponibles sur connecteurs qui permettent un empilement de cartes. Cartes Duemilanove, Uno, ...	Famille de cartes autour de μC 100 broches, avec des ATmega 1280 ou 2560 selon les modèles. Plus de ports sont disponibles sur connecteurs. 4 des connecteurs sont compatibles avec la famille de cartes à μC 28 broches. Carte MEGA, MEGA2560
	

Les platines Arduino peuvent être utilisées dans de nombreux domaines, dont la robotique. On peut utiliser divers outils de développement pour la création du programme exécutable. Avec certains outils, la programmation du μC cible s'effectue par une liaison USB, sans programmeur externe. On utilise alors l'autoprogrammation du μC cible avec l'aide d'un programme bootloader déjà enregistré en usine. *Voir plus loin pour plus de détails.*

Les principaux outils de développement sont :

- EDI Arduino gratuit. Programmation via une liaison USB.
- WinAVR + EDI tel que VMLAB ou AVRStudio gratuits
- FlowCode. Version d'évaluation gratuite. Version complète payante. Aucun programmeur n'est nécessaire (utilisation de la liaison USB)

<http://fr.wikipedia.org/wiki/Arduino> liste les différentes versions des cartes Arduino. *Certaines ne sont plus commercialisées.*

<u>Arduino</u>	Proces- seur	Flash KB	EE- PROM KB	SRAM KB	E/S lo- giques	...avec PWM	E/S lo- giques ou E ana.	Type d'interface USB	Dimen- sions mm
Diecimila	ATmega 168	16	0.5	1	14	6	6	FTDI	68.6mm x 53.3mm
Duemilanove	ATmega 168/328P	16/32	0.5/1	1/2	14	6	6	FTDI	68.6mm x 53.3mm
Uno	ATmega 328P	32	1	2	14	6	6	ATme- ga8U2	68.6mm x 53.3mm
Leonardo	ATmega 32U4	32	1	2,5	14	6	6	ATme- ga32U4	68.6mm x 53.3mm
Mega	ATmega 1280	128	4	8	54	14	16	FTDI	101.6mm x 53.3mm
Mega2560	ATmega 2560	256	4	8	54	14	16	ATme- ga8U2	101.6mm x 53.3mm
DUE	SAM3U4E	256	?	52	54	14	16	SAM3U4E	101.6mm x 53.3mm
Fio	ATmega 328P	32	1	2	14	6	8	Aucun	40.6mm x 27.9mm
Nano	ATmega 168 / 328	16/32	0.5/1	1/2	14	6	8	FTDI	43mm x 18mm
LilyPad	ATmega 168V / 328V	16	0.5	1	14	6	6	Aucun	50mm ø

Parmi les différentes familles de cartes, **la platine Arduino Uno avec un µC sur support est celle à retenir** pour de petites applications simples en STI2D SIN, car on peut changer le µC après destruction suite à une mauvaise manipulation élève. De plus l'adaptateur USB nécessaire pour la programmation du µC est intégré. Cette platine est compatible avec les autres de la même famille. C'est la seule qui est décrite ici.

De nombreuses cartes d'extension empilables sont disponibles dans le commerce : Wifi, LCD couleur, Ethernet, interface moteurs, etc.

Voir plus loin la partie consacrée aux cartes d'extension.

2 HISTORIQUE DU PROJET ARDUINO

http://fr.flossmanuals.net/arduino/ch002_historique-du-projet-arduino

Le projet Arduino est issu d'une équipe d'enseignants et d'étudiants de l'école de Design d'Interaction d'Ivrea¹ (Italie). Ils rencontraient un problème majeur à cette période (avant 2003 - 2004) : les outils nécessaires à la création de projets d'interactivité étaient complexes et onéreux (entre 80 et

¹ L'école « Interaction Design Institute Ivrea » (IDII) est aujourd'hui située à Copenhague sous le nom de « Copenhagen Institute of Interaction Design ».

100 euros). Ces coûts souvent trop élevés rendaient difficiles le développement par les étudiants de nombreux projets et ceci ralentissait la mise en œuvre concrète de leur apprentissage.

Jusqu'alors, les outils de prototypage étaient principalement dédiés à l'ingénierie, la robotique et aux domaines techniques. Ils sont puissants mais leurs processus de développement sont longs et ils sont difficiles à apprendre et à utiliser pour les artistes, les designers d'interactions et, plus généralement, pour les débutants.

Leur préoccupation se concentra alors sur la réalisation d'un matériel moins cher et plus facile à utiliser. Ils souhaitaient créer un environnement proche de Processing, ce langage de programmation développé dès 2001 par Casey Reas² et Ben Fry, deux anciens étudiants de John Maeda au M.I.T., lui-même initiateur du projet DBN³.

En 2003, Hernando Barragan, pour sa thèse de fin d'études, avait entrepris le développement d'une carte électronique dénommée Wiring, accompagnée d'un environnement de programmation libre et ouvert. Pour ce travail, Hernando Barragan réutilisait les sources du projet Processing. Basée sur un langage de programmation facile d'accès et adaptée aux développements de projets de designers, la carte Wiring a donc inspiré le projet Arduino (2005).

Comme pour Wiring, l'objectif était d'arriver à un dispositif simple à utiliser, dont les coûts seraient peu élevés, les codes et les plans « libres » (c'est-à-dire dont les sources sont ouvertes et peuvent être modifiées, améliorées, distribuées par les utilisateurs eux-mêmes) et, enfin, « multi-plates-formes » (indépendant du système d'exploitation utilisé).

Conçu par une équipe de professeurs et d'étudiants (David Mellis, Tom Igoe, Gianluca Martino, David Cuartielles, Massimo Banzi ainsi que Nicholas Zambetti), l'environnement Arduino est particulièrement adapté à la production artistique ainsi qu'au développement de conceptions qui peuvent trouver leurs réalisations dans la production industrielle.

Le nom Arduino trouve son origine dans le nom du bar dans lequel l'équipe avait l'habitude de se retrouver. Arduino est aussi le nom d'un roi italien, personnage historique de la ville « Arduin d'Ivrée », ou encore un prénom italien masculin qui signifie « l'ami fort ».

3 DOCUMENTATION SUR ARDUINO

Il existe un site « officiel » en français sur Arduino : <http://arduino.cc/fr/Main/HomePage>. Le site en anglais est plus complet. <http://www.arduino.cc/>

Un autre site propose beaucoup d'informations en français sur Arduino : <http://www.mon-club-elec.fr/>. Ce site est optimisé pour Firefox.

Une partie du site est une référence sur le langage de programmation Arduino et sur les bibliothèques fournies : http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php

L'autre partie du site est consacré à l'électronique programmée à base d'Arduino.

² Casey Reas était lui-même enseignant à IVREA, pour Processing, à cette époque.

³ Design By Numbers, un langage de programmation spécialement dédié aux étudiants en arts visuels.

On trouve de nombreux documents et sites web sur internet concernant Arduino. En voici quelques-uns :

http://sciences.siteduzero.com/tuto_pdf.php?vrsid=515602

Un document très progressif pour commencer, avec des rappels élémentaires d'électricité. Il peut être utilisé avec les élèves, éventuellement pour une autoformation pour les plus intéressés.

Le site web qui correspond : <http://sciences.siteduzero.com/tutoriel-3-515602-arduino-pour-bien-commencer-en-electronique-et-en-programmation.html>

<http://fr.flossmanuals.net/booki/arduino/arduino.pdf>

Livre réalisé par un collectif. En plus d'une approche progressive, une grande partie est consacrée à des exemples d'usage, notamment dans le domaine artistique. Un chapitre sur Scratch for Arduino.

Le site web qui correspond : <http://fr.flossmanuals.net/arduino/index>

<http://www.louisreynier.com/fichiers/KesacoArduino.pdf> présente la carte Arduino en quelques pages.

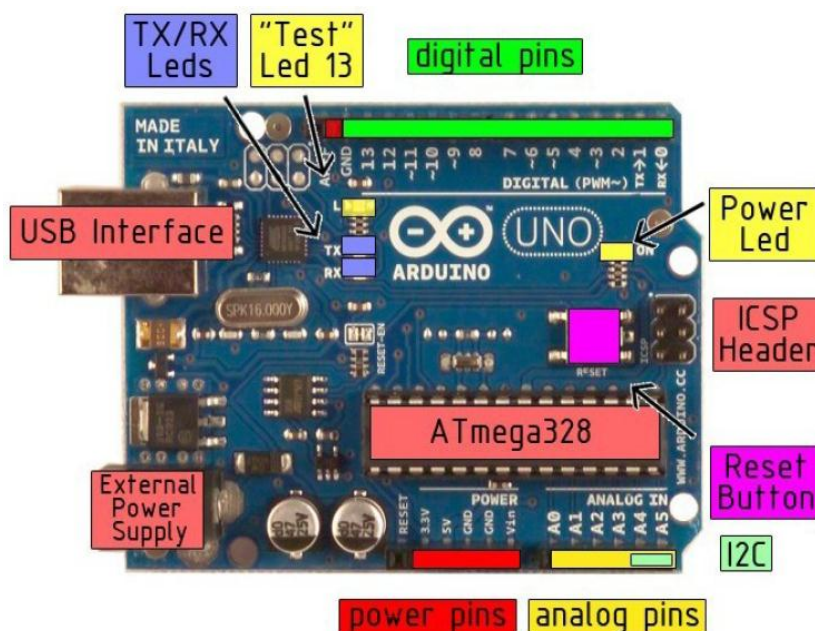
4 CARTES ARDUINO UNO

4.1 LES DIFFERENTES VERSIONS

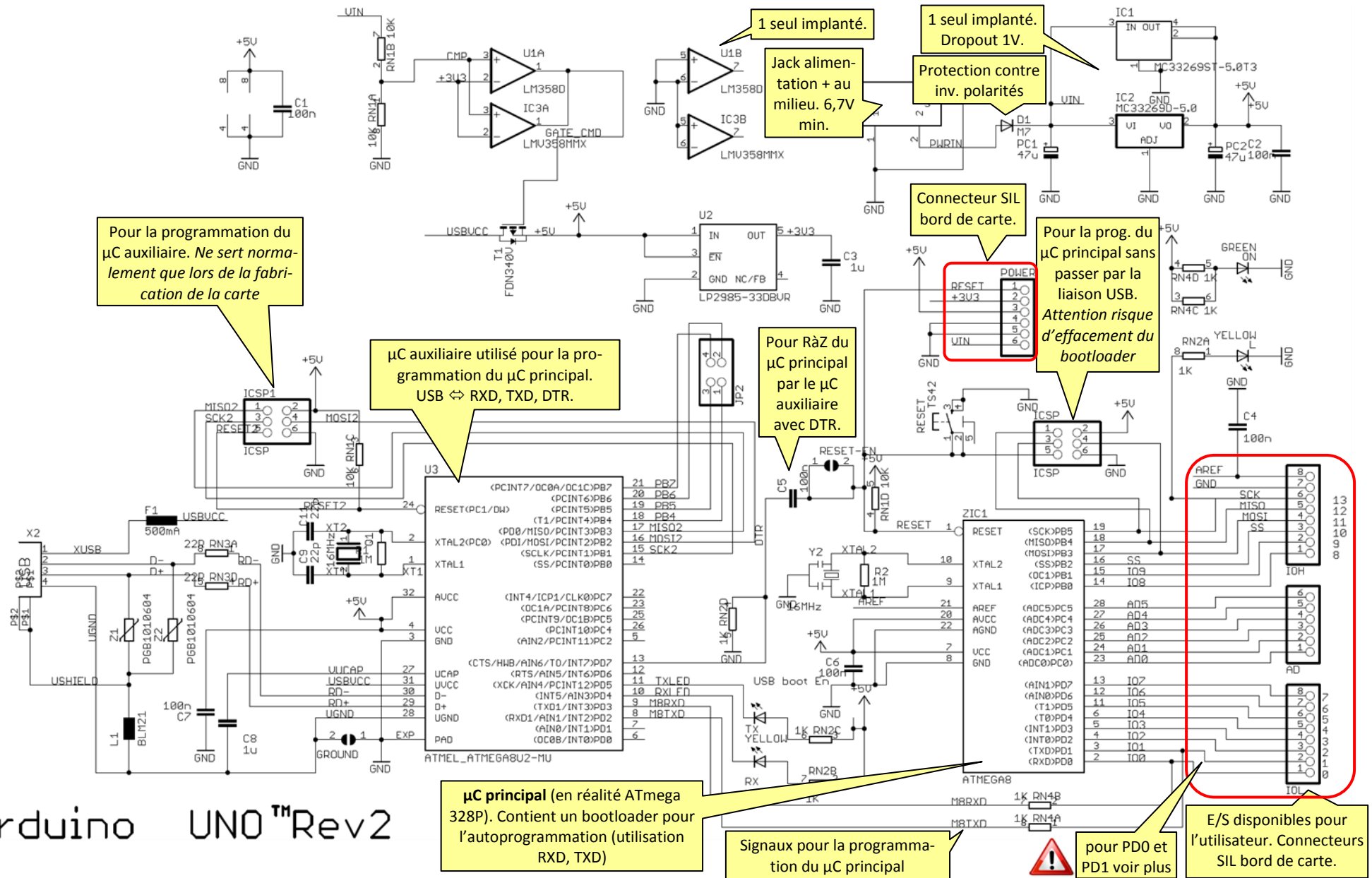
Il existe de petites différences entre les schémas et les implantations des différentes versions. Du point de vue de l'utilisateur, il n'y a aucune différence. Dans ce qui suit, on s'appuie sur l'Arduino Uno Rev 2 pour le schéma et l'implantation.

A la date de rédaction de ses lignes (février 2012), la dernière version est la rev 3.

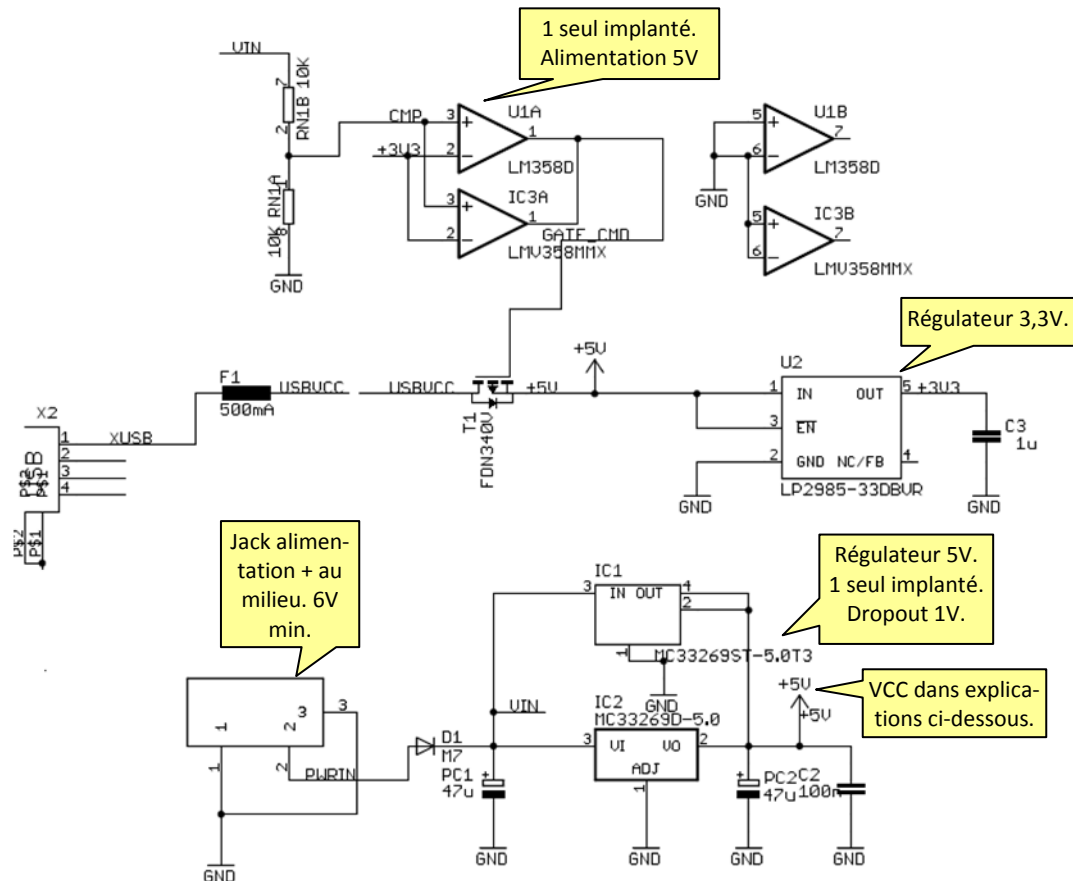
4.2 IMPLANTATION ET SCHEMA ARDUINO UNO REV2



Arduino UNO™ Rev2



4.3.1 SCHEMA



Si on connecte une carte interface de puissance Ardumoto avec un petit moteur d'un robot Pololu commandé en continu, on mesure VIN= 4,06V. Le moteur tourne.

Pb : si on alimente les 2 moteurs, le courant de démarrage risque de faire fondre le fusible F1.

Solution : utiliser une carte intermédiaire entre Arduino et Ardumoto. Ne pas connecter Vin de la carte Arduino et de la carte Ardumoto.

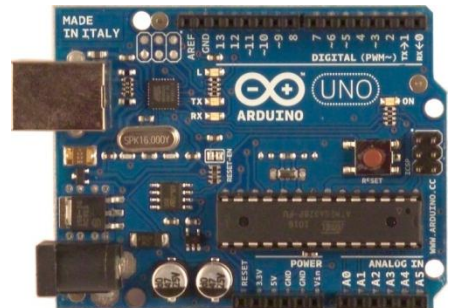
5 E/S DISPONIBLES SUR CONNECTEURS DE LA CARTE ARDUINO UNO

4 connecteurs SIL sont communs à toutes les versions. 3 de ces connecteurs sont utilisés pour les entrées / sorties. Ce sont les seuls décrits ici.

Toutes les broches peuvent être utilisées en E/S. Elles ont toutes au moins une autre fonction.



Pour PD0 et PD1. Voir plus loin.



Connecteur ANALOG IN

Nom Arduino	Broche μC	Autres fonctions	Remarque
A0	PC0	ADC0/PCINT8	
A1	PC1	ADC1/PCINT9	
A2	PC2	ADC2/PCINT10	
A3	PC3	ADC3/PCINT11	
A4	PC4	ADC4/SDA/PCINT12	liaison I2C
A5	PC5	ADC5/SCL/PCINT13	liaison I2C

Connecteurs DIGITAL (PWM ~)

Nom Arduino	Broche μC	Autres fonctions	Remarque
RX \leftarrow 0	PD0 	RXD / PCINT16	liaison série asynchrone. Aussi utilisée pour auto prog μC . R d'isolement de 1K pour cette fonction. <i>Voir plus loin pour la programmation du μC</i>
TX \rightarrow 1	PD1 	TXD/PCINT17	liaison série asynchrone. Aussi utilisée pour auto prog μC . R d'isolement de 1K pour cette fonction. <i>Voir plus loin pour la programmation du μC</i>
2	PD2	INT0/PCINT18	
~3	PD3	INT1/OC2B/PCINT19	Peut être utilisée pour générer des signaux dont PWM.
4	PD4	T0/XCK/PCINT20	
~5	PD5	T1/OC0B/PCINT21	Peut être utilisée pour générer des signaux dont PWM.
~6	PD6	AIN0/OC0A/PCINT22	Peut être utilisée pour générer des signaux dont PWM.
7	PD7	AIN1/PCINT23	
8	PB0	ICP1/CLKO/PCINT0	
~9	PB1	OC1A/PCINT1	Peut être utilisée pour générer des signaux dont PWM.
~10	PB2	/SS/OC1B/PCINT2	Peut être utilisée pour générer des signaux dont PWM.

Nom Arduino	Broche μ C	Autres fonctions	Remarque
			Liaison série synchrone SPI.
~11	PB3	MOSI/OC2A/PCINT3	Peut être utilisée pour générer des signaux dont PWM. Liaison série synchrone SPI. Est aussi utilisée pour la programmation ICSP.
12	PB4	MISO/PCINT4	Liaison série synchrone SPI. Est aussi utilisée pour la programmation ICSP.
13	PB5	SCK/PCINT5	Liaison série synchrone SPI. Est aussi utilisée pour la programmation ICSP. LED active à 1. R de 1K en série. Après RàZ, 3 clignotements rapides. Voir si gênant ou pas si on souhaite utiliser cette sortie. <i>Voir plus loin la partie sur la programmation.</i>



PD0 / PD1.

Avec la carte Arduino Uno Rev 2, le bootloader utilisé est Optiboot. La liaison série sert pour recevoir éventuellement des données dans le cas d'une programmation. Le bootloader valide donc la liaison série mais malheureusement il ne la désactive pas en fin d'exécution. Lorsque la liaison série est validée, il est impossible d'utiliser les broches PD0 et PD1 comme entrée ou sortie standard. Pour pouvoir utiliser ces broches comme simples E/S, l'utilisateur doit inhiber la liaison série en faisant `UCSR0B=0` ;

Voir la documentation du μ C ATmega328P.

Le comportement est-il identique avec d'autres versions du bootloader ?

6 ENVIRONNEMENTS DE DEVELOPPEMENT INTEGRES (EDI) UTILISABLES

6.1 EDI ARDUINO SPECIFIQUE A LA CARTE CIBLE

Cette partie est à réorganiser, car l'EDI Arduino va être décrit en détail en 2^{ème} partie de ce document.

6.1.1 GENERALITES

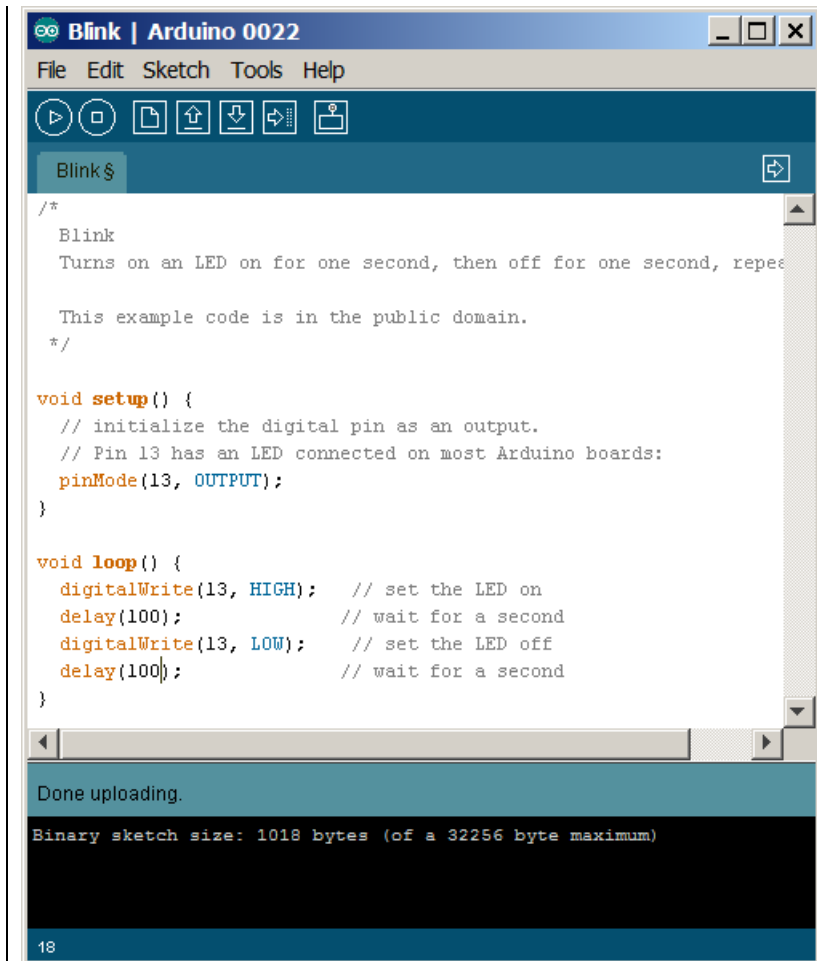
L'Environnement de Développement Intégré Arduino est très simple. Il se lance sans installation, à partir de l'exécutable arduino.exe.

Le langage de programmation est le langage C ou C++. De très nombreuses fonctions sont fournies. Leurs noms sont choisis pour une utilisation « intuitive ». Ex :
`digitalWrite(13, HIGH);`
écriture d'un 1 sur la sortie 13
13 est le numéro de la sortie repérée sur le connecteur.

Des exemples sont fournis dans le dossier exemple.

Le plus simple est dans le sous dossier Blink.

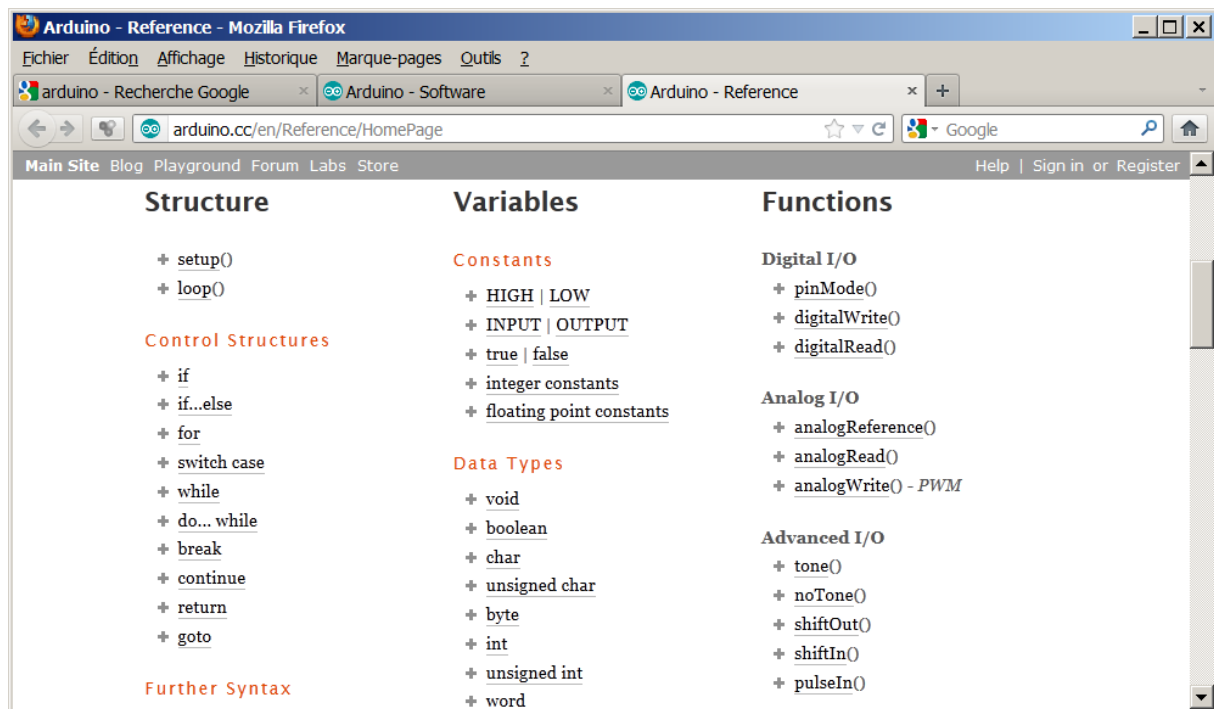
Dans cet exemple, le programme se compose uniquement d'une suite de fonctions.



Une icône permet de compiler le programme et de transférer l'exécutable dans le μ C cible via la liaison USB.

L'EDI et son utilisation sont décrites plus en détail sur la partie consacrée à l'EDI.

6.1.2 RAPPELS SUR LE LANGAGE / FONCTIONS DISPONIBLES

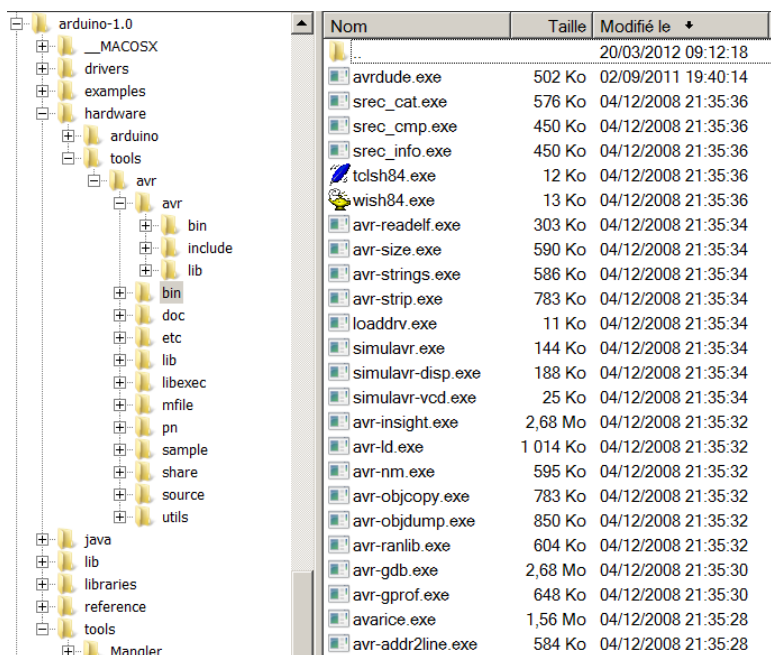


mon-club-elec.fr donne la même chose en français.

6.1.3 COMPILATION

La compilation s'appuie sur le compilateur gratuit AVR-GCC fourni avec le logiciel Arduino. Sa mise en œuvre est transparente à l'utilisateur.

Les fichiers d'AVR-GCC sont situés dans le dossier principal \ hardware \ tools \ avr \ bin

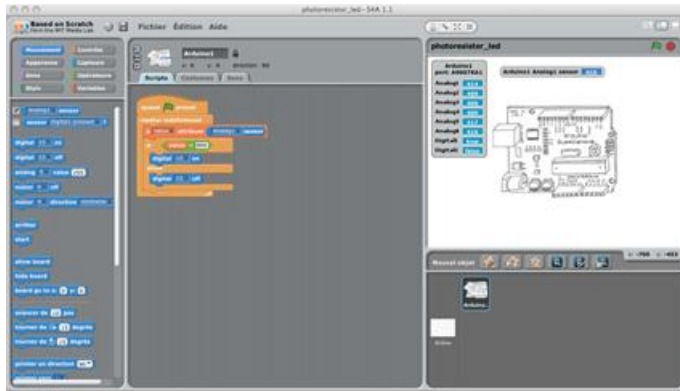


6.2 EDI SCRATCH FOR ARDUINO SPECIFIQUE A LA CARTE CIBLE

Cette EDI peut être utilisée avec des élèves de 2nde dans le cadre de l'option ISA.

S4A (Scratch For Arduino) est une extension (modification) de l'application Scratch qui a été créée en 2007 à des fins pédagogiques.

Cette extension permet de programmer des instructions que la carte d'interfaçage Arduino peut exécuter et ce depuis une GUI Scratch à laquelle les créateurs ont ajouté des fonctions spécifiques à la carte Arduino. En effet, le principe d'écriture d'un programme avec Scratch et S4A consiste à emboîter des blocs classés en code-couleur qui correspondent à des instructions programmatiques : fonctions, variables, opérations, lecture de fichiers importés (images, sons). S4A présente donc des nouveaux blocs destinés uniquement à une programmation d'interactivité entre des capteurs et des actionneurs via une carte Arduino.



Pour en savoir plus :

http://fr.flossmanuals.net/arduino/ch031_pour-aller-plus-loin

Une introduction avec des liens utiles

http://seaside.citilab.eu/scratch?_s=fuXmDa1SZk9_aSp9&_k=cHGDjGglnNohOpKR

Le site du projet Scratch for Arduino

Quelques projets :

<http://portic0312.qc.ca/robotique/spip.php?article117>

<http://chamayou.franck.free.fr/spip/spip.php?article252>

6.3 EDIS NON SPECIFIQUES A LA CARTE CIBLE

Pour des projets assez compliqué pour lesquels l'EDI Arduino n'est pas assez performant, on peut utiliser d'autres EDIs.

Avec certains, la programmation ne peut s'effectuer qu'avec un programmeur externe entre le PC et le connecteur 6 points ICSP de la carte. Attention au sens, car le connecteur n'a pas de détrompeur. On peut utiliser par exemple le programmeur AVRisp mkII.

Le risque de ce type de programmation est d'effacer le bootloader déjà enregistré en sortie d'usine.

Exemple d'EDI : CodeVision AVR (payant) / VMLAB + AVR-GCC (gratuits)/ Eclipse + AVR-GCC (gratuits) etc.

La page suivante liste les différents outils de développement utilisables.

<http://arduino.cc/playground/French/OutilsDeD%C3%A9veloppement>

Tous ne sont pas listés, loin de là.

Comme Arduino s'est énormément développé, de nombreux EDIs ont des greffons (plugins) qui permettent de s'adapter.

6.3.1 ECLIPSE

L'EDI Eclipse est gratuit. Il est prévu pour s'interfacer avec le compilateur AVR-GCC. Un plug-in permet de l'utiliser pour une cible Arduino.

Il existe plusieurs pages web qui expliquent comment utiliser Eclipse. Parmi celles-ci :

<http://www.arduino.cc/playground/Code/Eclipse>

<http://www.chipkin.com/articles/using-eclipse-with-arduino-duemilanove>

http://itpedia.nyu.edu/wiki/Arduino_in_Eclipse

<http://www.codeproject.com/Articles/110859/Arduino-Unleashed>

6.3.2 VISUAL MICRO = ARDUINO FOR VISUAL STUDIO (MICROSOFT)

<http://www.visualmicro.com/>

<http://www.visualmicro.com/page/Offer-Visual-Studio-Professional-Free-For-3-Years.aspx>

Pb : un lien est cassé pour l'inscription.

Passer par msdn ex : <http://msdn.microsoft.com/fr-fr/vstudio/default> puis connexion en haut à droite / Inscrivez-vous.

6.3.3 AVR STUDIO 5

<http://www.engblaze.com/tutorial-using-avr-studio-5-with-arduino-projects/>

6.4 FLOWCODE

FlowCode est un EDI non spécifique à Arduino.

Flowcode permet une édition graphique du programme source sous forme d'algorithme. La programmation du µC cible s'effectue via la liaison USB ou en utilisant un programmeur externe, selon un paramétrage effectué par l'utilisateur. Voir le document sur FlowCode.

6.5 LABVIEW

Il existe une interface LabView / Arduino <http://sine.ni.com/nips/cds/view/p/lang/fr/nid/209835>

- Accès aisé aux E/S numériques, entrées analogiques, PWM, I2C et SPI du microcontrôleur Arduino, à partir de LabVIEW
- Séquence (sketch) du moteur d'E/S à charger sur Arduino
- Exemples pour capteurs et tâches élémentaires
- Sans fil avec Bluetooth ou XBee
- Fréquences de boucles : par câble USB (200 Hz) et sans fil (25 Hz)
- L'ouverture de la séquence pour Arduino et les VIs du toolkit permettent de personnaliser sa fonctionnalité

7 PROGRAMMATION DU μ C PRINCIPAL PUIS UTILISATION

7.1 GENERALITES SUR LA PROGRAMMATION ET L'UTILISATION

Le μ C principal d'une carte Arduino peut se programmer de deux façons :

- Programmation In Situ (ICSP) avec un programmeur externe entre le PC et le connecteur 6 points ICSP de la carte. Attention au sens, car le connecteur n'a pas de détrompeur.
- Via la liaison USB. Le « bootloader » du μ C principal est utilisé. *Voir plus loin.*

Dans les 2 cas, le fichier à programmer, résultat d'une compilation, est au format normalisé .hex.

Après programmation, le programme utilisateur est lancé, suite à une RàZ ou mise sous tension, après le bootloader si aucune information arrive sur la liaison RXD.

Attention : *le bootloader utilisé (dernière version, Optiboot) fait clignoter 3 fois rapidement la LED L13 après RàZ pour indiquer qu'il attend des données sur la liaison série. Il faut donc éviter d'utiliser la sortie 13 (PB5) pour le cas où ces changements d'état après RàZ sont gênants. Ce défaut n'apparaît pas avec des versions plus anciennes du bootloader.*

7.2 PROGRAMMATION ICSP (In Circuit Serial Programming)

Le risque de ce type de programmation est d'effacer le bootloader si on écrit un programme sur ses adresses (voir l'annexe). Il faut un programmeur externe. N'importe quel outil de développement peut convenir. Par exemple CodeVision.

7.3 PROGRAMMATION VIA LA LIAISON USB

7.3.1 GENERALITES

La programmation par la liaison USB depuis l'EDI Arduino est très simple.

Sans passer par l'EDI Arduino, la programmation par la liaison USB a été difficile à mettre au point par l'auteur de ces lignes, car il a fallu beaucoup de temps pour trouver certaines informations. Une fois la mise au point réalisée, la programmation par la liaison USB est très pratique à mettre en œuvre, surtout avec des élèves. Ce qui suit donne toutes les informations nécessaires pour l'utilisation de la programmation par liaison USB.

7.3.2 PRINCIPE

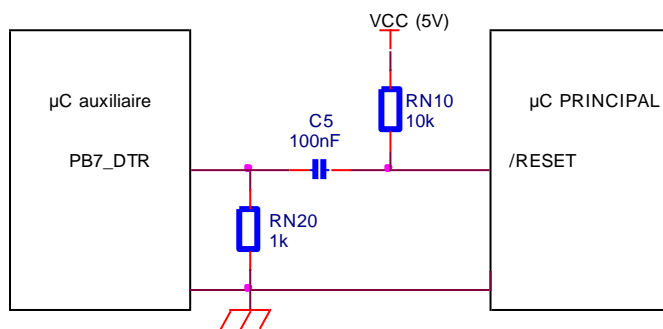
Dans l'Arduino Uno, un μ C auxiliaire réalise un interfaçage, aussi appelé un pont, USB / liaison série asynchrone 0V/5V, avec les signaux RXD, TXD et DTR.

Avec le pilote correctement installé sur le PC, la carte Arduino est vue par celui-ci comme un port COM (liaison série asynchrone).

Le PC effectue d'abord une RàZ du μ C principal de la carte Arduino en mettant DTR à 0. Ceci peut s'effectuer par exemple avec une ligne dans un fichier .bat. C'est la solution retenue par FlowCode :


```
@mode %ComPort%: baud=%BaudRate% parity=n data=8 stop=1 to=on xon=off odsr=off octs=off
dtr=on rts=off idsr=off
dtr=on
```

force à 0 la ligne DTR et provoque la RàZ du μ C principal de l'Arduino.



Après la RàZ, le μ C principal exécute le bootloader car il a été programmé pour cela. *Voir le document sur le bootloader.*

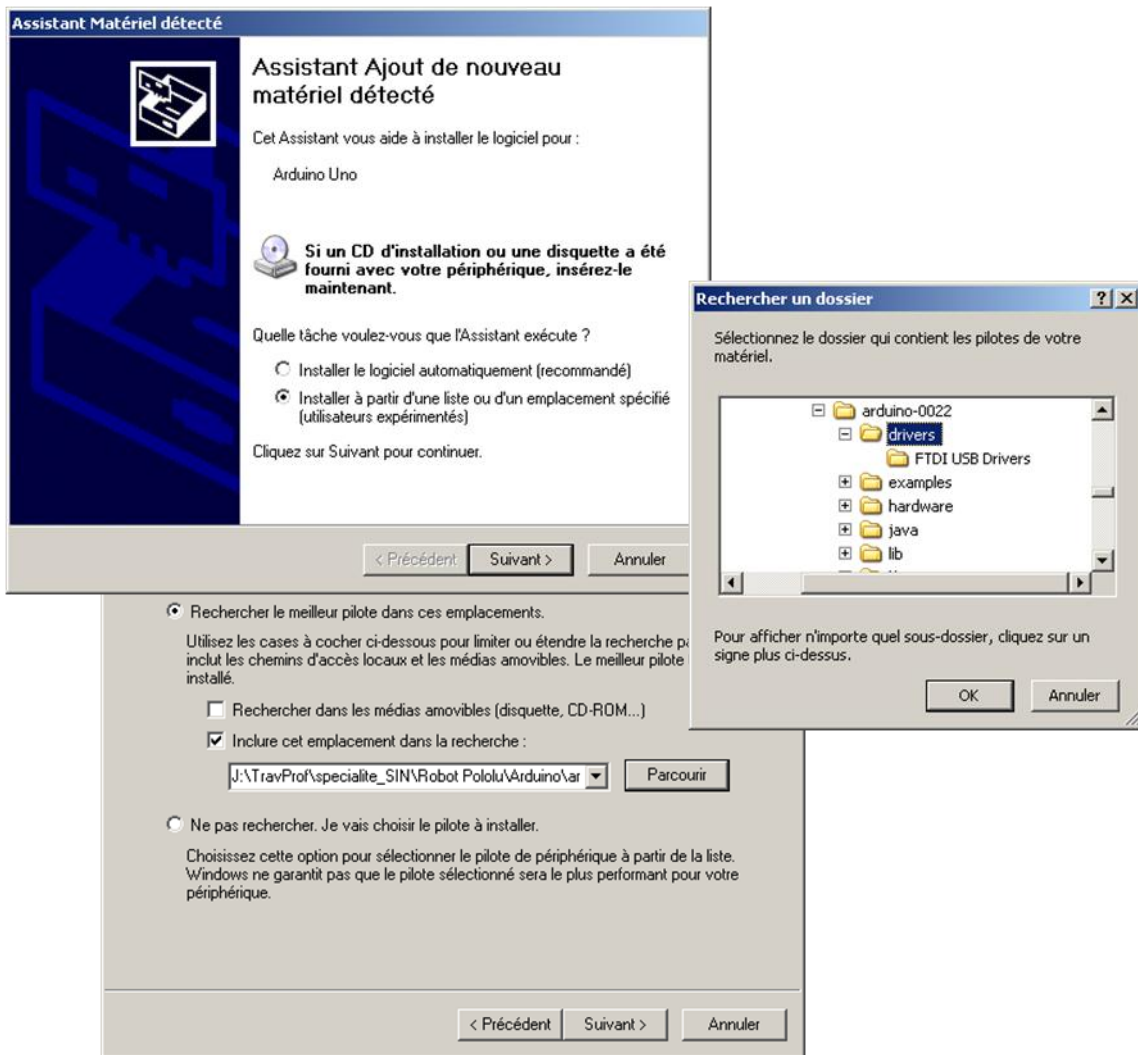
Le logiciel utilisé pour la programmation du μ C principal (Arduino, AVRdude de l'ensemble AVR-GCC, etc) envoie ensuite, avec un protocole bien défini (signaux TXD, RXD), des données normalement à destination d'un programmeur. Or c'est ici le μ C principal lui-même qui se 'fait passer' pour un programmeur. Ainsi, il récupère les données du programme pour s'autoprogrammer.

Le μ C principal utilise pour ce faire un programme particulier qu'il contient déjà : un bootloader.

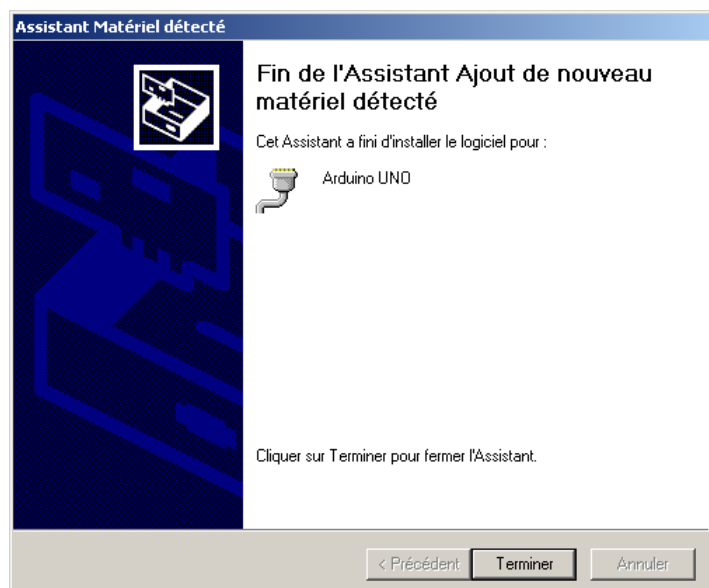
7.3.3 INSTALLATION DU PILOTE USB

Il faut d'abord récupérer le pilote puis l'installer. Pour récupérer le pilote, il faut télécharger la dernière version du logiciel Arduino (1.0 à la date de rédaction de ces lignes) et la décompacter. Le pilote est dans le dossier drivers : ArduinoUNO.inf, Arduino UNO REV3.inf, etc.

Lorsqu'on connecte pour la première fois le câble USB, il faut spécifier le dossier qui contient le pilote.



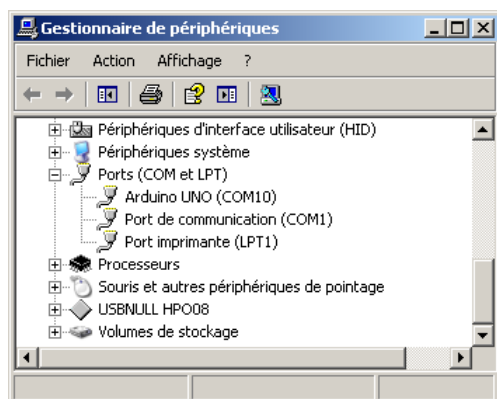
La recherche et l'installation prennent quelques instants.



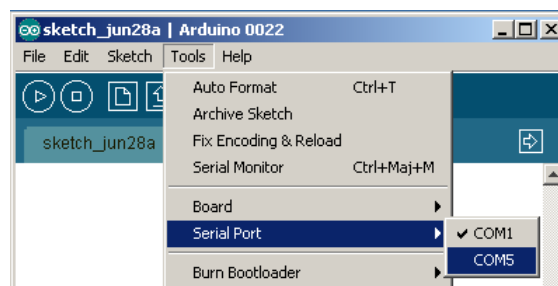
7.3.4 CARTE ARDUINO VUE DU LOGICIEL DE PROGRAMMATION

Une fois le pilote correctement installé, la carte Arduino est vue comme un port COM. Le numéro du port varie d'un ordinateur à l'autre.

Exemple :



La carte est vue comme un port COM10.
Le nom de la carte s'affiche directement dans le gestionnaire de périphérique. Uniquement avec carte Uno rev2 ou +



La carte est vue comme un port COM5

7.3.5 PROGRAMMATION DEPUIS FLOWCODE

FlowCode utilise AVRdude. Pour désigner le μ C et le fichier .hex, on peut utiliser des caractères de substitution, ce qui évite d'avoir à modifier la ligne de commande entrée dans une boîte de dialogue. Il faut effectuer avant de lancer AVRdude, une RàZ du μ C principal. Ceci est effectué avec une ligne d'un fichier .bat. Voir le document sur FlowCode

7.3.6 PROGRAMMATION AVEC AVRDUDE

7.3.6.1 Téléchargement d'AVRdude

Le logiciel AVRdude (AVR **D**ownloader/**U**pload**E**r) qui fait partie de AVR-GCC est un logiciel gratuit et performant. Ce logiciel peut être lancé après configuration d'une ligne de commande depuis plusieurs logiciels (FlowCode, etc).

Attention : certaines versions d'AVRdude ne fonctionnent pas avec la dernière version du bootloader. AVRdude 5.10 ou plus fonctionne correctement.

Pour récupérer la dernière version d'AVRdude, on peut télécharger WinAVR qui contient, un compilateur, un éditeur de liens, ... et le logiciel de programmation AVRdude.

<http://sourceforge.net/projects/winavr/files/>

Après installation de WinAVR, avrdude.exe et avrdude.conf sont dans le dossier d'installation \bin. Si WinAVR n'est pas utilisé par ailleurs, copier ces fichiers dans un dossier puis supprimer l'installation de WinAVR. Récupérer avant \doc\avrdude\avrdude.pdf.

On peut aussi récupérer AVRdude.exe dans les dossiers d'Arduino. Chemin d'accès : Les fichiers dossier principal \ hardware \ tools \ avr \ bin. Arduino 1.0 intègre la dernière version d'AVRdude. *Ce n'était pas le cas avec les versions précédentes d'Arduino.*

7.3.6.2 Ligne de commande AVRdude

Exemple de ligne de commande pour un μ C ATmega328, une liaison USB vers une carte UNO vue comme une liaison com8. *Cette ligne de commande doit suivre dans un fichier « batch » la RàZ du μ C principal avec la commande mode présentée plus haut.*

```
avrdude -p m328 -c arduino -P com8 -U flash:w:"NomFichier.hex"
```

Le chemin d'accès de NomFichier.hex n'est pas nécessaire si avrdude est lancé depuis le dossier parent.

Pour le détail des commandes, voir le manuel avrdude. Ce manuel est dans le dossier d'installation de WinAVR\doc\avrdude.

On peut aussi le trouver à <http://savannah.spinellcreations.com/avrdude>

Il est aussi dans un dossier d'Arduino. Chemin d'accès : Dossier principal \ hardware \ tools \ avr \ doc \ avrdude. *La version du manuel n'est pas la dernière avec Arduino 1.0. Elle ne correspond pas à la version de l'exécutable fourni.*

8 CARTES D'EXTENSION (SHIELDS)

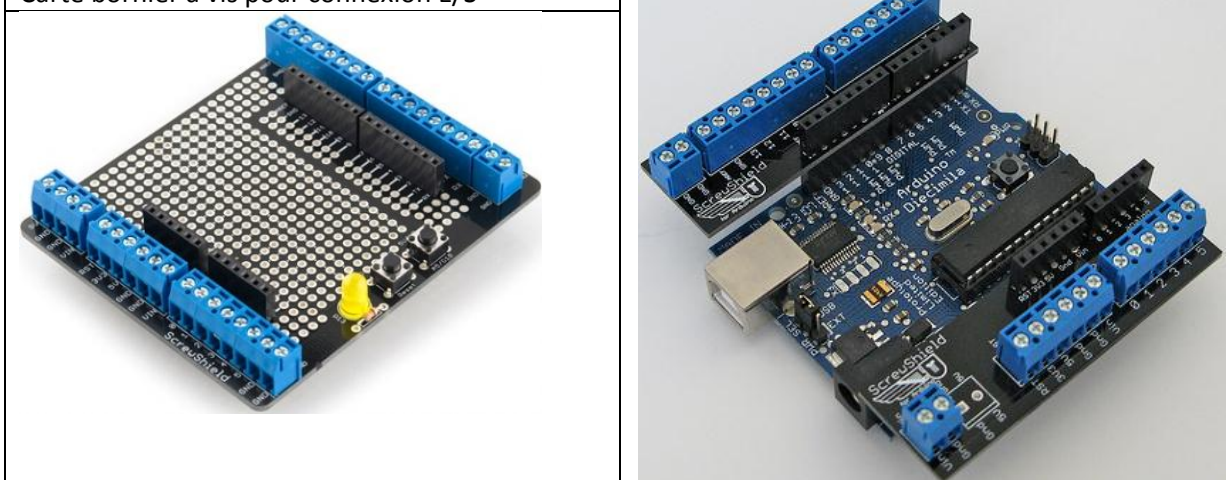
Shield signifie bouclier. Ce terme a certainement été retenu car la carte d'extension s'enfiche sur la carte Arduino et constitue une espèce de bouclier.

Un fabricant de carte d'extension utilise le terme Mezzanine.

La page suivante liste plusieurs cartes Arduino et des cartes compatibles, avec les revendeurs :

http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.MATERIEL

Carte bornier à vis pour connexion E/S

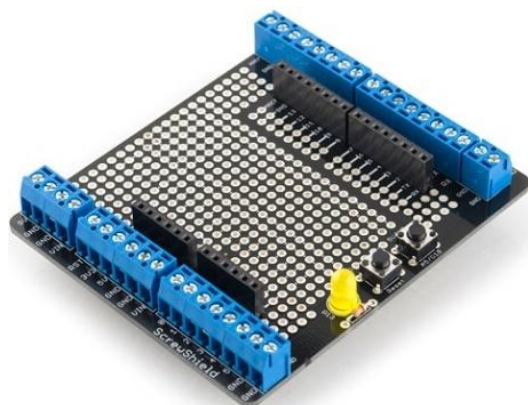


Quelques distributeurs (certains n'acceptent les commandes que par Internet)

Lextronic. http://www.lextronic.fr	http://www.lextronic.fr/R2478-arduino.html Grand choix dont une carte avec borniers. Commandes lycées acceptées
Snootlab http://shop.snootlab.com	http://shop.snootlab.com/6-arduino Grand choix de cartes dont une avec borniers. Commandes ly-

	cées acceptées.
http://fr.hobbytronics.co.uk/	Vente uniquement par internet. Assez grand choix dont carte avec borniers.
http://www.jlectronique.org	Carte avec borniers.
http://www.zartronic.fr	

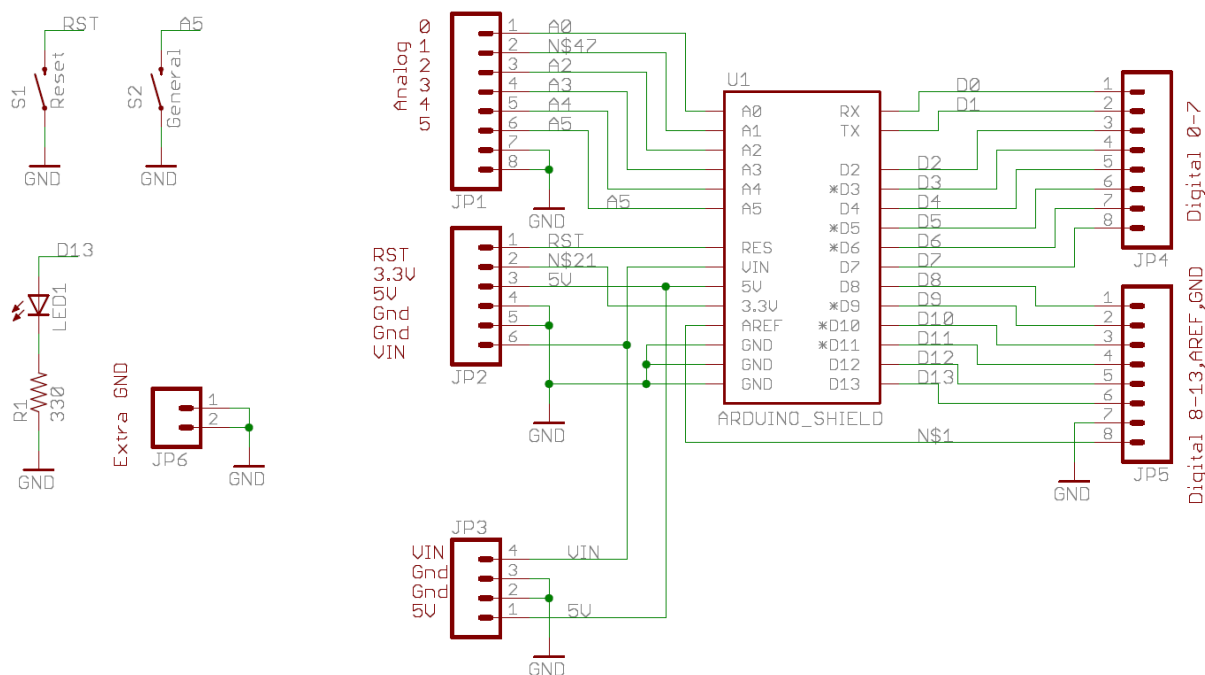
8.1 CARTES AVEC ZONE DE CABLAGE (+ BORNIER)

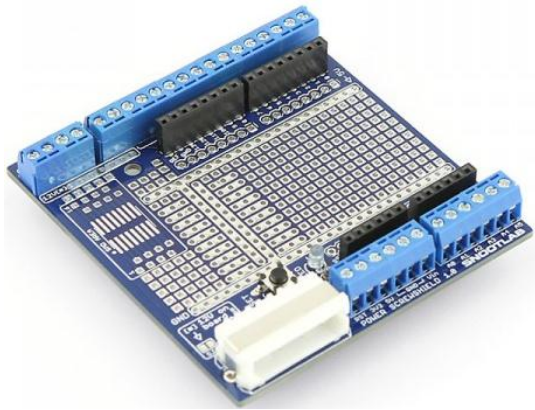


Disponible chez Lextronic

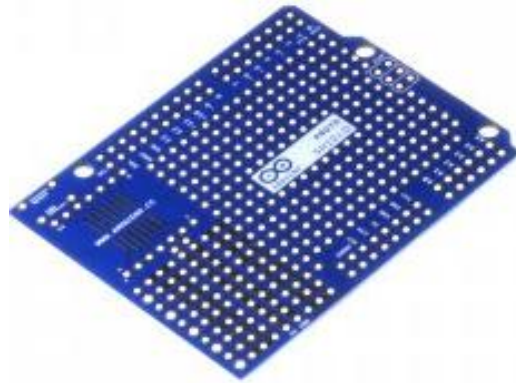
Référence : **DEV-09729**

12.76 € HT **15.26 € TTC**





Disponible chez Snootlab
Référence : KIT-00501
14,95 € TTC



Proto PCB Rev3. Disponible sous forme de carte seule ou de module avec les connecteurs. Pas de borniers.

Par exemple, chez Farnell :

Carte seule. ref 2075379 5,35 € HT

Module. ref 2075345 13,95 € HT

8.2 CARTE AFFICHEUR LCD LIAISON // LCD_KEYPAD



Vendue par Zartronic

<http://www.zartronic.fr/shield-lcd-pour-arduino-p-125.html>

Prix : 17,00€

Modèle : DFR0009

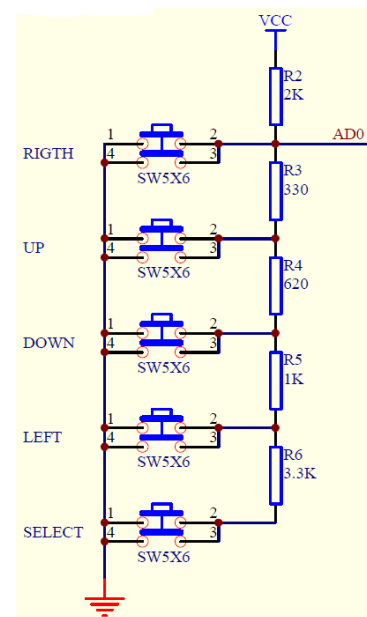
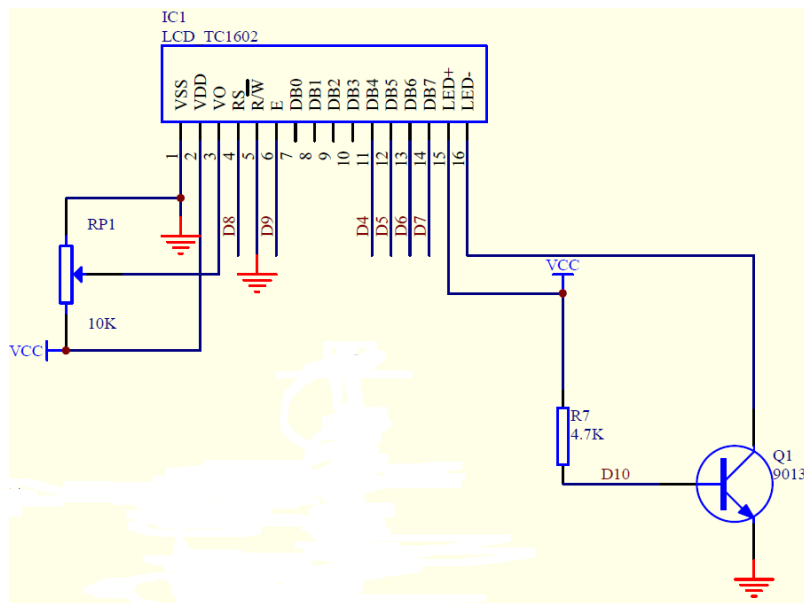
En Commande >10 jours

Fabricant : DFRobot

Commandes lycées certainement non acceptées.

Courriel envoyé le 21/12/2012 à Lextronic pour savoir s'ils peuvent s'en procurer. Réponse : produit non RoHS. Interdit d'importation.

Schéma



Pour minimiser le nombre de broches utilisés, les BP sont réunis à une seule entrée analogique. Si plusieurs BPs sont simultanément appuyés, seul le plus haut sur le schéma est pris en compte.

Nom carte	Nom Arduino	Broche μC	Autres fonctions	Remarque
AD0	A0	PC0	ADC0/PCINT8	
D4	4	PD4	T0/XCK/PCINT20	DB4
D5	~5	PD5	T1/OC0B/PCINT21	DB5
D6	~6	PD6	AIN0/OC0A/PCINT22	DB6
D7	7	PD7	AIN1/PCINT23	DB7
D8	8	PB0	ICP1/CLKO/PCINT0	RS
D9	~9	PB1	OC1A/PCINT1	E

Compatibilité Arduino + Flowcode : oui.

Compatibilité Arduino + Carte intermédiaire (voir plus loin) + Ardumoto + Flowcode : Non sans modif carte intermédiaire version 1. Oui si modif carte intermédiaire version 1. Il faut utiliser la sortie 11 de la carte Arduino (PB3) et la connecter à D9 (signal E).

8.3 AFFICHEUR LCD LIAISON I2C



Vendu en kit par snootlab.

<http://shop.snootlab.com/shields-snootlab/135-deuligne.html>

Référence : KIT-01051

24,00 € TTC

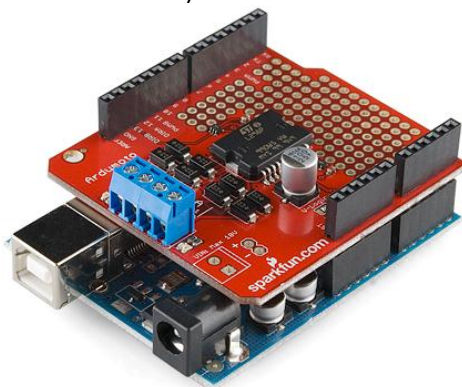
L'utilisation d'une liaison I2C fait que l'afficheur ne peut être commandé directement par Flowcode car il n'existe pas de « macros » pour un afficheur I2C. L'utilisateur doit écrire ses propres « macros ».

8.4 INTERFACE 2 MOTEURS ARDUMOTO

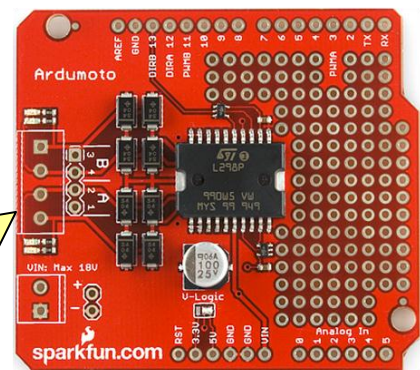
Cette petite platine permet de piloter 2 moteurs (2A max). Des LEDs CMS bleues et jaunes permettent d'indiquer le sens de rotation des moteurs.

Référence Lextronic : **DEV-09815**

22.46 € TTC (18.78 € HT) (la carte est vendue sans les connecteurs).



Vers les moteurs (bornier ou connecteur à souder)



Pour plus de détail : <http://www.sparkfun.com/products/9815>

Le cœur de la carte est un CI spécialisé L298. Le L298 est constitué de 4 demi ponts qui sont utilisés ici pour réaliser 2 ponts en H, avec inversion des signaux entre IN1 et IN2 et entre IN3 et IN4.

PWMx commande la validation d'un pont. DIRx permet de commander la direction.

3 cas sont possibles.

Fonctionnement déduit du schéma de la carte et de la notice technique du CI utilisé :

PWMn	DIRn	
0	X	Moteur non alimenté par le pont. La tension aux bornes du moteur est proche de 0V si une des diodes de roue libre conduit ou si le moteur est à l'arrêt.
1	0	tension aux bornes du moteur est égale à $\approx +V_{IN}$
1	1	tension aux bornes du moteur est égale à $\approx -V_{IN}$

La tension de déchet est de l'ordre de 1,2V pour un courant de 0,8A

Il n'est pas possible de commander un freinage rapide du moteur.

Nom carte	Nom Arduino	Broche μ C	Autres fonctions	Remarque
PWMA	~3	PD3	INT1/OC2B/PCINT19	PWM avec timer 2
DIRA	12	PB4	MISO/PCINT4	
PWMB	~11	PB3	MOSI/OC2A/PCINT3	PWM avec timer 2
DIRB	13	PB5	SCK/PCINT5	

Compatibilité Flowcode : non.

Flowcode génère des signaux PWM en OC1A et OC1B. Solution : carte d'interface qui redirige les signaux. Cette carte peut aussi comporter des borniers pour connexions de capteurs, etc.

Compatibilité LCD_KeyPad : oui. Aucune entrée/sortie en commun.

Voir plus loin le § sur la carte intermédiaire.

8.5 MOTOPROTO SHIELD MODKIT

Cette carte est fabriquée par Sparkfun (sparkfun.com)

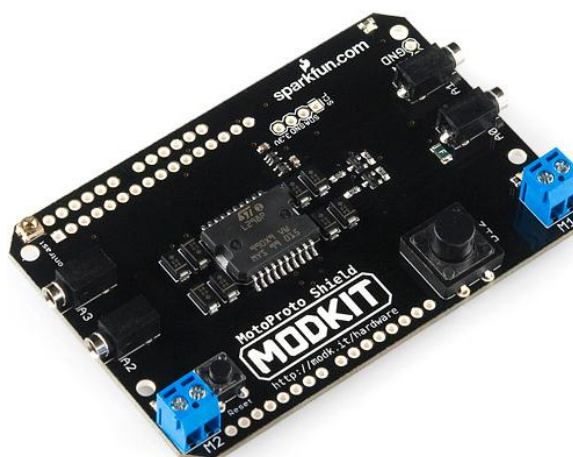
Cette carte a plusieurs fonctionnalités :

- Raccordement de 4 capteurs via des prises jack
- Interface de petite puissance pour 2 moteurs courant continu (tels que ceux utilisés dans des robots)
- Connexion d'un afficheur LCD standard 2 lignes de 16 caractères.

Elle est vendue sans les connecteurs et sans l'afficheur.

La carte et les accessoires sont disponibles chez Lextronic.

Réf carte DEV-10018 28,70 € TTC



8.5.1 BROCHES UTILISEES

Commande moteurs

Nom Modkit	Nom Arduino	Broche μ C	Autres fonctions	Remarque
DIRA	2	PD2	INT0/PCINT18	
PWMA	~3	PD3	INT1/ OC2B /PCINT19	non compatible Flowcode
DIRB	4	PD4	T0/XCK/PCINT20	
PWMB	~5	PD5	T1/ OC0B /PCINT21	non compatible Flowcode

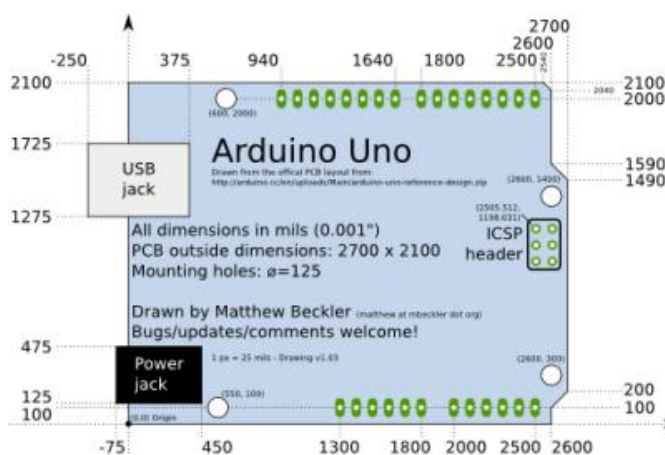
Les 2 sorties PWM utilisées ne sont pas issues du même timer, ce qui complique inutilement la programmation dans des cas simples, par exemple deux moteurs sur un robot.

Pour une utilisation avec Flowcode, il faut une carte intermédiaire spécifique qui redirige les signaux.

9 DIMENSIONS DE LA CARTE, EMPLACEMENT DES CONNECTEURS

9.1 CARTE COTEE

Les dimensions sont données d'après le fichier `arduino_uno_drawing_500x351.png` disponible sur Internet.



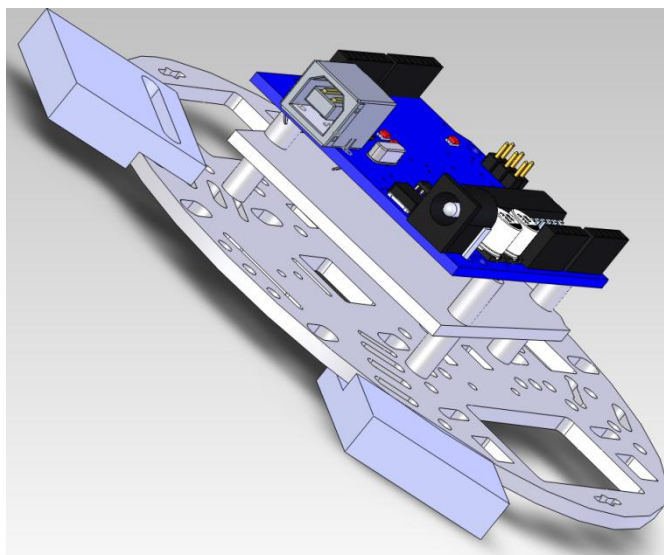
9.2 FICHIERS SOLIDWORKS

On trouve sur internet les fichiers Solidworks pour réaliser des assemblages.

<http://grabcad.com/library/arduino-uno-reference-design>

Le fichier s'appelle « Arduino UNO Final.zip »

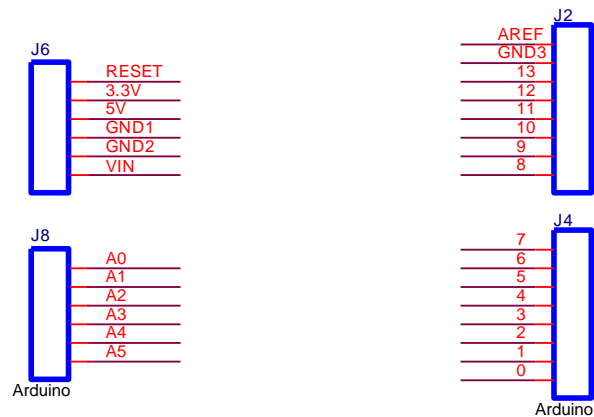
Ci-contre, un assemblage réalisé par un élève au lycée Pierre Emile Martin à Bourges, dans le cadre d'un projet.



9.3 MODELES ORCAD POUR CREER DES CARTES D'EXTENSION

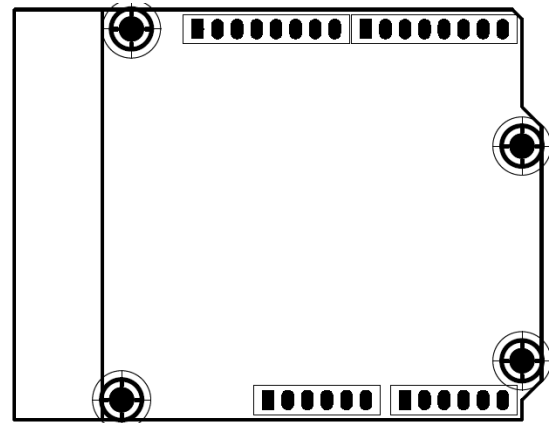
Le projet Connect_Arduino (Lycée Pierre Emile Martin à Bourges) correspond au schéma des 4 connecteurs empilables de la carte.

Il permet de réutiliser ces connecteurs par copier/coller dans un nouveau schéma.



Le fichier Contour Arduino.max correspond à la carte avec les 4 connecteurs ci-dessus. Il peut être utilisé comme modèle pour la création d'une nouvelle carte.

Les connecteurs correspondent à ceux du schéma.



10 CARTE D'INTERCONNEXION PEM

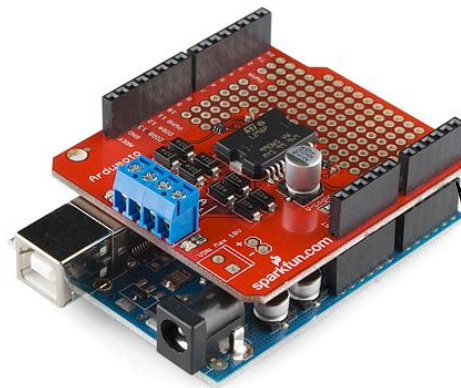
10.1 GENERALITES

Cette carte est conçue et fabriquée au lycée Pierre Emile Martin.

Cette carte a plusieurs fonctions :

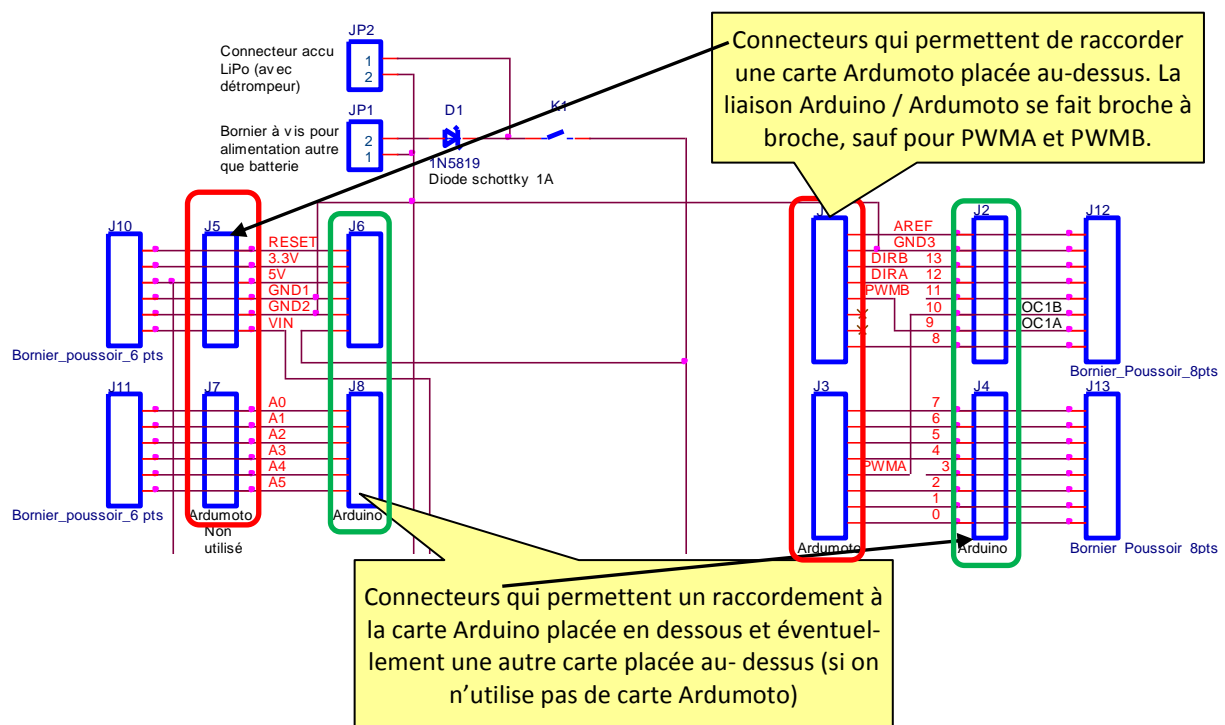
- Connexion alimentation (batterie d'accumulateurs) + Interrupteur M/A général
- Circuit de protection pour la batterie LiPo
- LEDs de visualisation (batterie OK, tension insuffisante)
- Modification des connexions pour la PWM entre Arduino et Ardumoto pour utilisation avec FlowCode (voir la partie sur FlowCode).
- Borniers pour la connexion de capteurs

Cette carte possède 2 séries de connecteurs qui permettent de superposer une carte avec toutes les connexions Arduino ou d'insérer une carte Ardumoto en léger décalage avec les connexions compatibles FlowCode pour la commande.



Sans carte intermédiaire, les cartes s'emboîtent directement avec liaison directe entre les connecteurs

Le schéma partiel de la carte est donné ci-après :



10.2 SIGNAUX GENERES PAR LA CARTE ARDUINO AVEC FLOWCODE

FlowCode ne permet pas de générer des signaux PWM sur les sorties de la carte Arduino utilisées par la carte Ardumoto de commande des moteurs. La carte intermédiaire permet de rediriger les sorties Arduino comme suit :

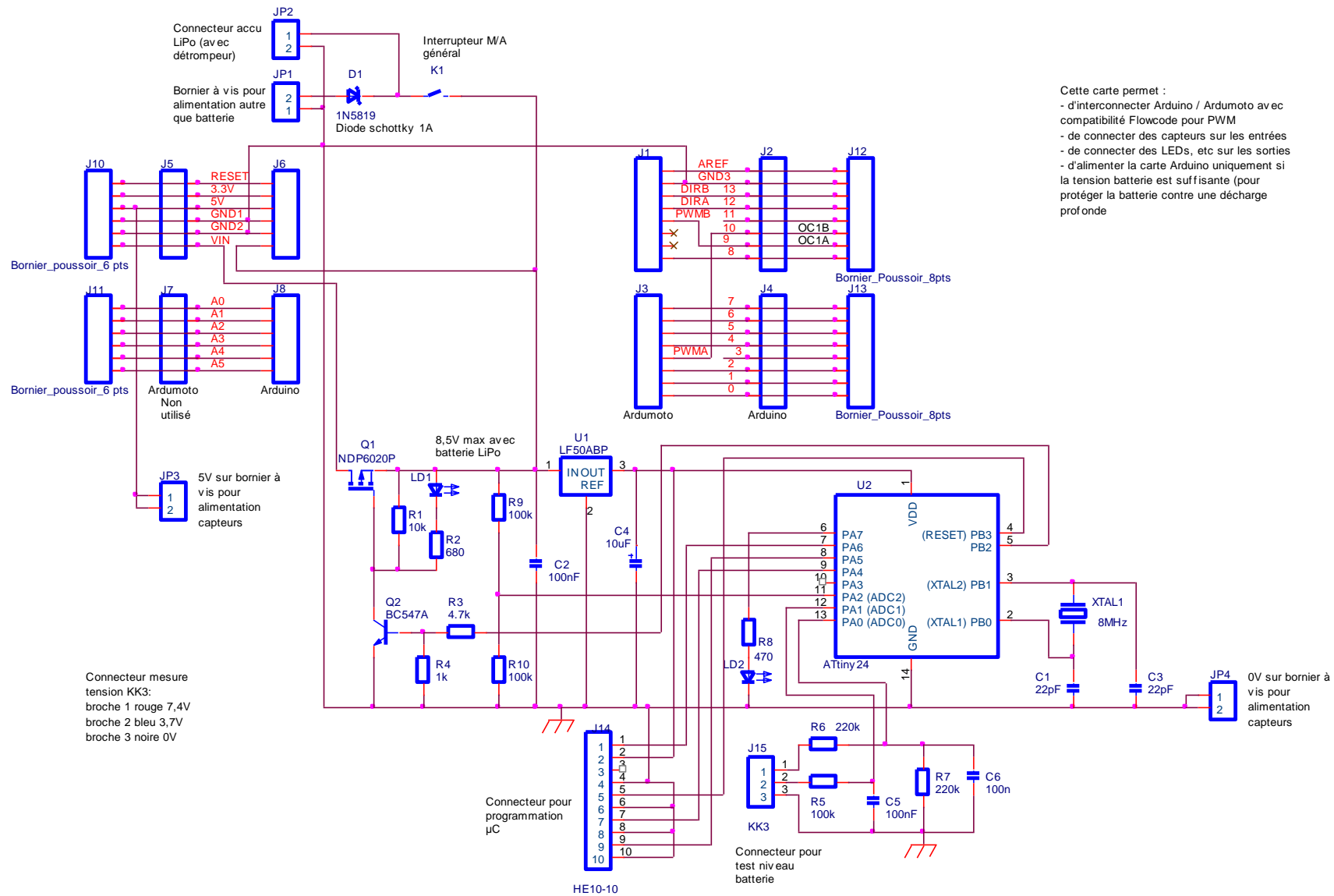
Sortie carte Arduino	Broche du µC	Nom Flowcode	Entrée carte Ardumoto
9 (broche 2 connecteur IOH)	PB1 (OC1A)	PWM canal 1	PWMB (broche 11)
10 (broche 3 connecteur IOH)	PB2 (OC1B)	PWM canal 2	PWMA (broche 3)
12	PB4	B4	DIRA (broche 12)
13	PB5	B5	DIRB (broche 13)

Remarque : il aurait été plus simple de réaliser les connexions suivantes :
9 Arduino (PWM1) → 3 Ardumoto (PWMA)

10 Arduino (PWM2) → 9 Ardumoto (PWMB)

10.3 SCHEMA COMPLET

Voir page suivante :



Cette carte permet :

- d'interconnecter Arduino / Arduimoto avec compatibilité Flowcode pour PWM
- de connecter des capteurs sur les entrées
- de connecter des LEDs, etc sur les sorties
- d'alimenter la carte Arduino uniquement si la tension batterie est suffisante (pour protéger la batterie contre une décharge profonde)

10.4 NOTES DE CONCEPTION

10.4.1 CAN

Réf CAN 5V / CAN utilisé sur 10 bits / $LSB = V_{ref} / 2^{10} = 4,88 \text{ mV}$

$N = V_e / V_{ref} / 2^{10}$

Capacité d'échantillonnage / blocage du CAN : 14 pF. Si le circuit d'attaque a une résistance équivalente de 100 K Ω (cas pour la voie 1) \rightarrow Constante de temps = 1,4 μS . Pour obtenir une précision de 1%, il faut attendre au moins 5 τ avant de lancer la CAN après avoir choisi la voie à convertir.

Avec un convertisseur 10 bits, pour obtenir le maximum de précision, il faut attendre bcp plus. On retient 100 μS .

10.4.2 ALIMENTATIONS

L'alimentation générale des cartes Arduino et Ardumoto se fait à partir de la carte intermédiaire sur laquelle est directement branchée la batterie.

Si la tension délivrée par la batterie est trop faible, l'alimentation Vin de la carte Ardumoto est coupée pour éviter de trop décharger la batterie. La carte Arduino est toujours alimentée. Sa consommation est faible. Un voyant clignote pour indiquer de recharger la batterie.

Cette solution a été retenue suite au fonctionnement constaté pour l'alimentation de la carte Arduino décrit ci-dessous.

La carte Arduino peut être alimentée uniquement par le câble USB qui sert à la programmation, lorsque l'interrupteur de la carte intermédiaire est sur la position Arrêt.

Sur la carte Arduino, si $V_{PWRIN} = 0$, le régulateur IC2 (ou IC1 selon implantation) est utilisé à l'envers. Il conduit en sens inverse et délivre une tension VIN.

Valeurs relevées : VIN = 4,29V si aucune autre carte d'extension (shield) n'est connectée.

Si on connecte sur la carte intermédiaire la carte interface de puissance Ardumoto avec un petit moteur d'un robot Pololu commandé en continu, on mesure VIN = 4,06V. Le moteur tourne.

Pb : si on alimente les 2 moteurs, le courant de démarrage risque de faire fondre le fusible F1.

La solution est de séparer les Vin des cartes Arduino et Ardumoto.

Toujours lorsque la carte Arduino est uniquement alimentée à partir de la liaison USB, La tension VIN délivrée par la carte Arduino d'environ 4,2V permet au régulateur +5V de la carte intermédiaire de délivrer une tension proche de 4,2V au μC qui fonctionnent. Il faut dans ce cas tester la tension Vin et si elle est trop faible, il faut bloquer l'alimentation Vin de la carte Ardumoto.

10.4.3 TENSION DELIVREE PAR LA BATTERIE ET SEUILS

	Tension nominale (à mi charge environ)	Tension max pleine charge	Seuil pour autoriser le fonctionnement	Seuil min pour arrêter le fonctionnement
1 élément	3,7V	4,2V	3,55V	3,3V

2 éléments	7,4V	8,4V	7,1V	6,6V
------------	------	------	------	------

Il manque des infos pour choisir les seuils assez finement. Aucune doc trouvée pour la capacité restante en fonction de la tension à vide. Les seuils sont estimés en fonction des courbes de décharge avec le courant le plus faible (pour notre utilisation, courant de décharge < 1C). Voir le document sur les batteries LiPo.

Tension	3,55V	3,3V	2,5V		
Résultat CAN Avec Vr _{éf} = 5V	727	675	512		

10.4.4 CONNECTEURS

Connecteurs à utiliser pour les batteries :

Désignation	Ref constructeur	Ref Farnell
Embase 2 contacts Tyco type PE	AMP - 1586041-2 - EMBASE CI COUDEE 2 VOIES UL94V-2	811-5974
Boîtier femelle Tyco PE2	AMP - 794954-2 - BOITIER FEMELLE 2 VOIES 94V-2	811-6261

Documents pour la fabrication :

Schéma OrCAD : Carte_Intermediaire2.opj + Carte_Intermediaire2.dsn

Routage OrCAD : CARTE_INTERMEDIAIRE2B.MAX

11 CARTE LCD PEM

A la date de rédaction de ces lignes (mars 2012), il n'existe pas encore de carte LCD disponible correspondant aux besoins. Lextronic doit bientôt en proposer.

11.1 CONNEXIONS ET COMPATIBILITES

La carte LCD décrite est compatible avec l'association carte Arduino + carte Ardumoto pour les 3 EDIs suivants : Flowcode, Arduino, CodeVisionAVR.

La compatibilité avec les EDIs Arduino et Flowcode est normale car dans les 2 cas, il est possible d'affecter indépendamment chacune des broches du LCD à une broche quelconque du µC.

Pour Arduino, voir <http://arduino.cc/fr/Main/LiquidCrystal>.

Pour Flowcod, voir doc perso.

FlowCode n'utilise pas le bit RW du LCD dans ses macros.

Avec CodeVisionAVR, le câblage du LCD est imposé comme suit :

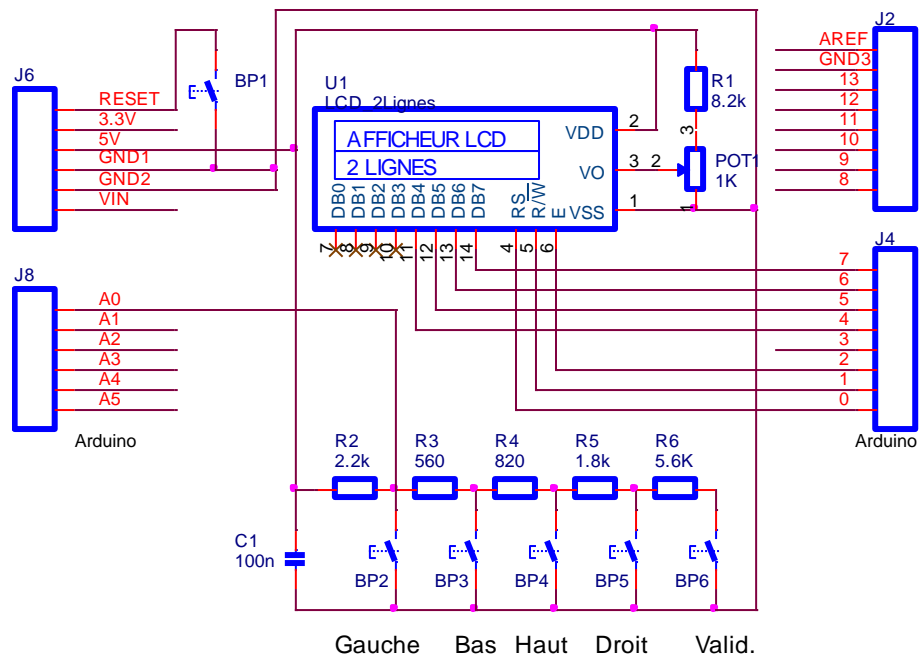
LCD	RS	RD ou RW	EN		DB4	DB5	DB6	DB7
N° broche LCD	4	5	6		11	12	13	14
Bit d'un Port µC	0	1	2	3	4	5	6	7

En tenant compte des broches disponibles avec l'assemblage carte Arduino + carte interconnexion PEM + carte Ardumoto, le câble retenu est le suivant :

Repères des broches sur le connecteur Arduino	0	1	2	3	4	5	6	7
Broche µC	PD0	PD1	PD2	PD3	PD4	PD5	PD6	PD7
LCD	RS	RW	EN		DB4	DB5	DB6	DB7

La carte Ethernet utilise les broches 10, 11, 12, 13. Il est peut être possible de l'utiliser en même temps que le LCD. Si on utilise la carte Ethernet, il ne faut pas valider le LCD. Voir s'il y a une validation pour l'utilisation de la carte Ethernet.

11.2 SCHEMA STRUCTUREL



11.3 TENSION SELON APPUI BP

Si plusieurs BPS sont simultanément appuyés, seul celui le plus à gauche sur le schéma est pris en compte.

Valeurs des résistances :

$R_2 = 2,2K$ Valeur choisie arbitrairement dans la série E12.

BP3 appuyé, on veut $VA_0 \approx 1V$

$$1 = 5 * R_3 / (R_2 + R_3) \rightarrow R_2 = 4 R_3 \rightarrow R_3 = R_2 / 4 = 550\Omega \rightarrow \text{valeur normalisée } 560\Omega$$

BP4 appuyé, on veut $VA_0 \approx 2V$

$$2 = 5 * (R_3 + R_4) / ((R_2 + R_3) + R_4) \rightarrow 2 R_2 - 3 R_3 = 3 R_4 \rightarrow R_4 = 2/3 R_2 - R_3 = 906 \Omega \rightarrow \text{valeur normalisée } 820 \Omega$$

BP5 appuyé, on veut $VA_0 \approx 3V$

$$3 = 5 * (R_3 + R_4 + R_5) / ((R_2 + R_3 + R_4) + R_5) \rightarrow 3 R_2 - 2 R_3 - 2 R_4 = 2 R_5 \rightarrow R_5 = 3/2 R_2 - R_3 - R_4 = 1920 \Omega \rightarrow \text{valeur normalisée } 1800 \Omega$$

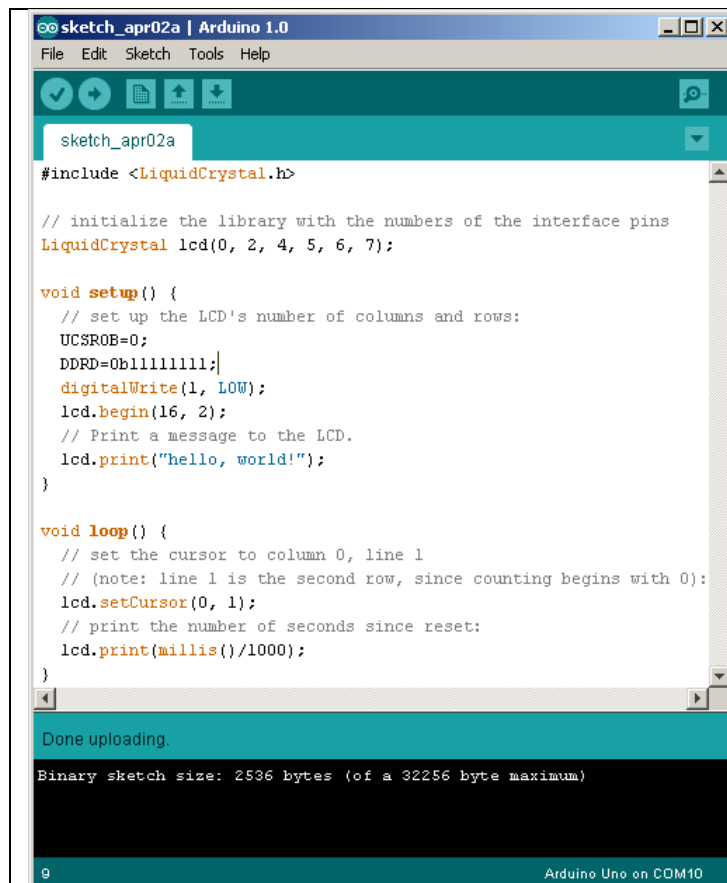
BP6 appuyé, on veut $VA_0 \approx 4V$

$$4 = 5 * (R_3 + R_4 + R_5 + R_6) / ((R_2 + R_3 + R_4 + R_5) + R_6) \rightarrow 4 R_2 - R_3 - R_4 - R_5 = R_6 = 5620 \rightarrow \text{valeur normalisée } 5,6 K \Omega$$

BP appuyé	Aucun	BP2	BP3	BP4	BP5	BP6
Tension approximative en A0 (V)	5	0	1	2	3	4

11.4 UTILISATION AVEC L'EDI ARDUINO

Voir remarque § 5 E/S disponibles sur connecteurs de la carte Arduino Uno.

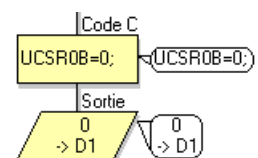


La même méthode `lcd.print` est utilisée pour afficher du texte à partir d'une chaîne de caractères et un nombre à partir d'une variable.

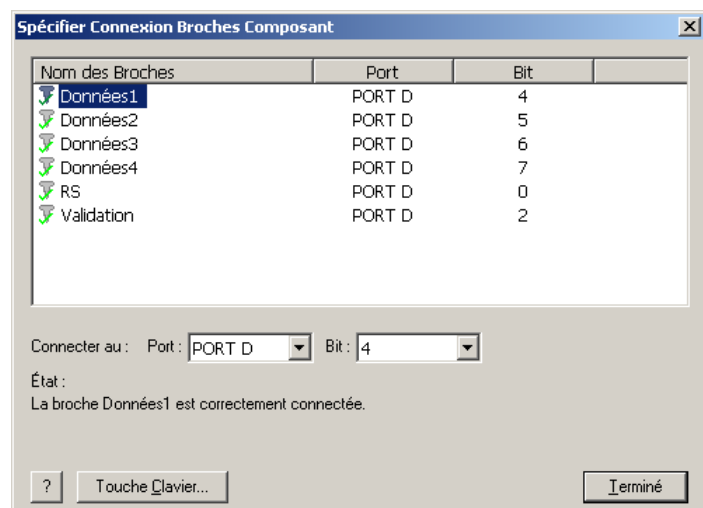
11.5 UTILISATION AVEC FLOWCODE

Voir remarque § 5 E/S disponibles sur connecteurs de la carte Arduino Uno.

A cause du problème mentionné plus haut avec la carte Arduino Uno, il faut ajouter une boîte Code C dont le contenu est montré ci-contre. Flowcode ne gère pas la broche R/W. Comme elle est connectée au bit PD1, il faut le forcer à 0.



Les connexions du composant LCD doivent être fixées comme suit :



12 SCHEMATISATION ET FABRICATION DE CARTE D'EXTENSION AVEC FRITZING

Il existe un logiciel qui permet de dessiner une carte de connexion rapide avec des composants interconnectés et reliés aux E/S d'une carte Arduino. On peut ensuite obtenir le schéma et router une carte d'extension.

Le tout est relativement facile à mettre en œuvre et peut être utilisé par un élève dans le cadre d'un projet STI2D SIN à condition que le nombre de composants soit extrêmement limité.

<http://fritzing.org>

Il existe des vidéos et 4 tutoriels en français au format pdf :

<http://fritzing.org/learning/translations/>

13 QUELQUES EXEMPLES D'UTILISATION D'ARDUINO

13.1 ROBOTIQUE

Il existe de très nombreuses utilisations d'Arduino en robotique. Seules quelques-unes sont mentionnées ici.

13.1.1 ROBOTS NXT DE LEGO

Il existe une carte d'interface entre une carte Arduino et les capteurs et actionneurs d'un robot NXT. Des bibliothèques de fonctions sont disponibles.

La carte est vendue par génération robot. Il existe certainement d'autres revendeurs.

<http://www.generationrobots.com/shield-pour-arduino,fr,3,68.cfm>



Le prix est exorbitant (87,50 €) car il ne s'agit que d'interconnexions. Voir si d'autres distributeurs proposent la carte moins chère.



Pour une information technique complète, voir :

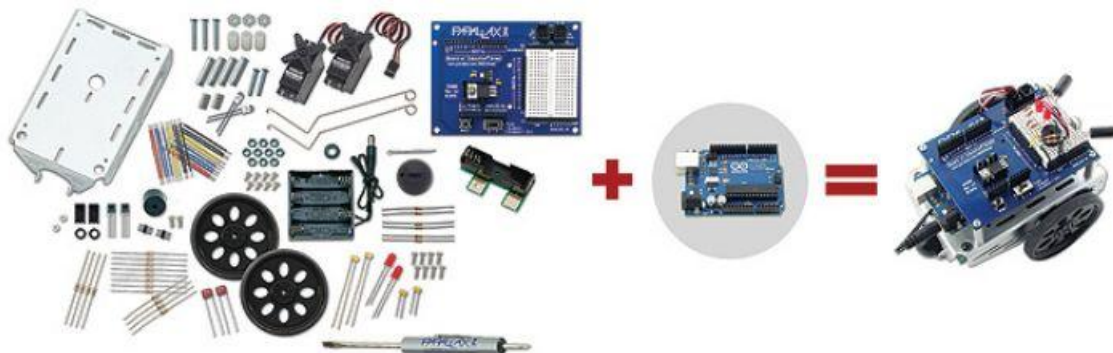
http://www.openelectrons.com/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=7

13.1.2 BOE SHIELD BOT

Le Boe Shield Bot est un robot mobile programmable qui s'appuie sur une carte Arduino. Il est ainsi programmable à l'aide de l'environnement de programmation Arduino et possède une planche de connexion sans soudure vous permettant facilement d'y ajouter vos composants électroniques, capteurs et vos propres circuits électroniques.

Pour plus de détails, voir par exemple :

<http://www.generationrobots.com/robot-mobile-boe-shield-pour-arduino-parallax,fr,4,Robotics-Shield-kit.cfm>



La carte supplémentaire est fabriquée par Parallax. Appellation: Board of Education Shield.
<http://www.parallax.com/BOEShield>

13.1.3 POLOLU

Il est possible d'adapter une carte Arduino + une carte Ardumoto + une carte d'interconnexion PEM (par exemple) à un châssis Pololu. L'avantage de ce dernier est son faible coût. L'ensemble est vendu par Lextronic. Ce sont les références de ce revendeur qui sont données ci-dessous :

Composition :

- Un disque en matière plastique prédécoupé
- 2 moteurs/réducteurs "CC" (Réf.: [MOT-993](#))
- Un support de fixation pour les moteurs (Réf.: [POL1089](#))
- Une paire de roue (Réf.: [POL1090](#))
- Une roue folle type "Ball-Caster" (Réf.: [ROB08909](#))



Des trous correspondent à deux des trous de la platine Arduino.

L'écart entre les deux moteurs est d'environ 29 mm. L'expérience a montré qu'il faut placer suffisamment de poids à l'avant au niveau de la bille pour éviter que le robot bascule en arrière lors d'une forte accélération. *Il peut être utile de placer une deuxième roue folle à l'arrière, avec une différence de niveau par rapport à l'avant.*

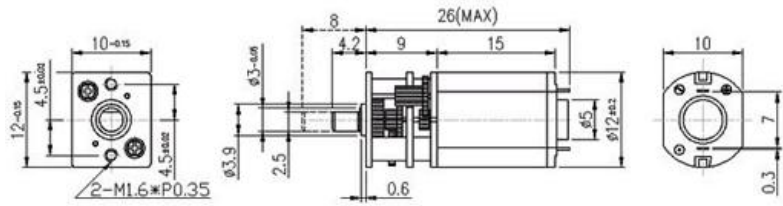
Référence Lextronic du châssis complet : **POL1506** **32.00 € TTC** (prix fin 2011)

Moteur réducteur miniature "POLOLU 993"

Ce moteur réducteur est très recherché par ses petites dimensions et sa grande qualité. Doté d'un rapport de réduction de 30:1 et de pignons en métal, il dispose d'un arbre en "D".



- Dimensions: 24 x 10 x 12 mm
- Poids: 0,34 oz
- Réduction: 30:1
- Alimentation 3 à 7 V env.
- Vitesse à vite @ 6 V: 440 rpm
- Courant à vide: 40 mA
- Courant en charge: 360 mA
- Couple: 4 oz.in



oz : once, ounce en anglais. 1 oz = 1/16 lb (livre, pound en anglais) = 28,3g.

ozf : ounce-force = 1 oz * g = 0,278 N. C'est ce qu'il faut retenir pour le couple. Le f a été omis.

rpm : rotation per minute, tour par minute.

Attention : le bloc réducteur est assez fragile. Lors de l'insertion des roues sur l'axe, il faut appuyer très fort. Il faut absolument appuyer à l'arrière du moteur en face des entretoises du bloc réducteur.

13.2 PROJETS DISPONIBLES SUR INTERNET

<http://hacknmod.com/hack/top-40-arduino-projects-of-the-web/>

<http://www.instructables.com/tag/type-id/category-technology/channel-arduino/>

EDI ARDUINO

1 LANGAGE DE PROGRAMMATION

Notes à remettre en forme

Derrière Arduino il y a le compilateur C/C++ AVR-GCC.

Il supporte toutes les constructions standards du langage C et quelques-uns des outils du C++
Arduino ajoute une surcouche pour que la syntaxe utilisateur soit la plus simple possible.

Un programme est constitué de 2 fonctions :

void setup() // la partie initialisation

void loop() // la boucle sans fin

2 ARDUINO ET LINUX

Il est possible de faire fonctionner Arduino sous GNU/Linux.

Le détail est donné sur cette page :

[http://www.mon-club-
elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.DebuterInstallationLinux](http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.DebuterInstallationLinux)

3 FONCTIONNEMENT DE LA COMPILATION

Le fichier édité dans Arduino a l'extension pde (avant la version 1.0) ou ino (depuis 1.0). Lorsqu'on lance la compilation, il est d'abord traduit en fichier .cpp puis ensuite les différents exécutables de la suite AVR-GCC sont appelés.

Lors de la compilation les fichiers générés sont rangés dans
User\NomUtilisateur\AppData\Local\Temp\buildxxx.xx.tmp\

Par rapport au fichier .ino, des lignes sont rajoutées au début du fichier .cpp, après les #include utilisateur et les commentaires, avant la 1^{ère} instruction :

#include "Arduino.h" // avec Arduino 1.0 ; avant c'était #include « WProgram.h »

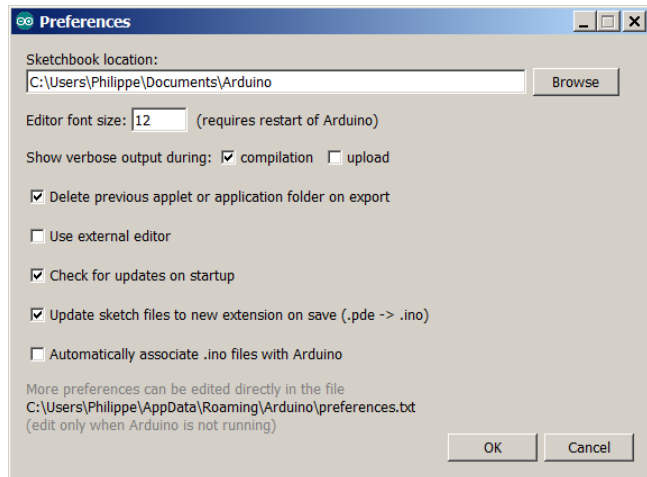
void setup(); // prototype pour chacune des fonctions utilisées

void loop();

Le fichier Arduino.h est rangé dans Dossier_Arduino\hardware\arduino\cores\arduino.

Il contient les inclusions des fichiers en-tête des bibliothèques usuelles d'AVR-GCC, des définitions de macros, des équivalences symboliques, des en-têtes de fonctions qui sont dans des fichiers .c fournis avec Arduino, l'inclusion de pins_arduino.h. Ce dernier est dans Dossier_Arduino\hardware\arduino\variants\ eightanaloginputs ou leonardo ou ... Ce fichier en inclut lui-même d'autres.

Pour voir ce qui se passe lors de la compilation, on peut cocher l'option « Show verbose output during compilation ». La boîte de dialogue est ouverte avec File / Preferences.



Lors de la compilation lancée depuis l'EDI, le fichier source .cpp est traduit en fichier objet .o, de même que le ou les fichiers éventuellement inclus par l'utilisateur.

Les fichiers .c ou .cpp du Dossier_Arduino\hardware\arduino\cores\arduino sont traduits en fichier objet .o puis intégrés dans la bibliothèque core.a (commande : avr-ar rcsv).

Ces opérations sont refaites à chaque lancement d'une compilation depuis l'EDI. On ne voit pas l'intérêt. La bibliothèque core.a aurait pu être livrée toute prête et ensuite utilisée.

Les fichiers objets .o issus du fichier source principal et des éventuels fichiers inclus par l'utilisateur sont liés avec les fonctions utilisées issues de la bibliothèque core.a et le fichier .hex est créé.

Pour plus de détail, voir :

<http://openhwareplatform.blogspot.fr/2011/03/inside-arduino-build-process.html>

PROCESSING

interface graphique programmable côté PC pour le système Arduino

Voir http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.OUTILSProcessing