# ARDUINO

## LES CARTES ARDUINO ET LEURS UTILISATIONS

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 Ardumofo .............................................................................. 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 :

<table>
<thead>
<tr>
<th>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, ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>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</td>
</tr>
</tbody>
</table>


Les principaux outils de développement sont :

- EDI Arduino gratuit. Programmation via une liaison USB.
- WinAVR + EDI tel que VMLAB ou AVRStudio gratuits

<table>
<thead>
<tr>
<th>Arduino</th>
<th>Processeur</th>
<th>Flash KB</th>
<th>EE-PROM KB</th>
<th>SRAM KB</th>
<th>E/S logiques...avec PWM</th>
<th>E/S logiques ou E ana.</th>
<th>Type d'interface USB</th>
<th>Dimensions mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diecimila</td>
<td>ATmega 168</td>
<td>16</td>
<td>0.5</td>
<td>1</td>
<td>14</td>
<td>6</td>
<td>6</td>
<td>FTDI</td>
</tr>
<tr>
<td>Duemilanove</td>
<td>ATmega 168/328P 16/32</td>
<td>0.5/1</td>
<td>1/2</td>
<td>14</td>
<td>6</td>
<td>6</td>
<td>FTDI</td>
<td></td>
</tr>
<tr>
<td>Uno</td>
<td>ATmega 328P</td>
<td>32</td>
<td>1</td>
<td>2</td>
<td>14</td>
<td>6</td>
<td>6</td>
<td>ATmega8U2</td>
</tr>
<tr>
<td>Leonardo</td>
<td>ATmega 32U4</td>
<td>32</td>
<td>1</td>
<td>2,5</td>
<td>14</td>
<td>6</td>
<td>6</td>
<td>ATmega32U4</td>
</tr>
<tr>
<td>Mega</td>
<td>ATmega 1280</td>
<td>128</td>
<td>4</td>
<td>8</td>
<td>54</td>
<td>14</td>
<td>16</td>
<td>FTDI</td>
</tr>
<tr>
<td>Mega2560</td>
<td>ATmega 2560</td>
<td>256</td>
<td>4</td>
<td>8</td>
<td>54</td>
<td>14</td>
<td>16</td>
<td>ATmega8U2</td>
</tr>
<tr>
<td>DUE</td>
<td>SAM3U4E</td>
<td>256</td>
<td>?</td>
<td>52</td>
<td>54</td>
<td>14</td>
<td>16</td>
<td>SAM3U4E</td>
</tr>
<tr>
<td>Fio</td>
<td>ATmega 328P</td>
<td>32</td>
<td>1</td>
<td>2</td>
<td>14</td>
<td>6</td>
<td>8</td>
<td>Aucun</td>
</tr>
<tr>
<td>Nano</td>
<td>ATmega 168 / 328</td>
<td>16/32</td>
<td>0.5/1</td>
<td>1/2</td>
<td>14</td>
<td>6</td>
<td>8</td>
<td>FTDI</td>
</tr>
<tr>
<td>LilyPad</td>
<td>ATmega 168V / 328V</td>
<td>16</td>
<td>0.5</td>
<td>1</td>
<td>14</td>
<td>6</td>
<td>6</td>
<td>Aucun</td>
</tr>
</tbody>
</table>

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


Le projet Arduino est issu d'une équipe d'enseignants et d'étudiants de l'école de Design d'Interaction d'Ivrea\(^1\) (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 Copenhagen 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 matériau moins cher et plus facile à utiliser. Ils souhaitaient créer un environnement proche de Processing, ce langage de programmation développé dès 2001 par Casey Reas et Ben Fry, deux anciens étudiants de John Maeda au M.I.T., lui-même initiateur du projet DBN.


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

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 consacrée 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 :

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.

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


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
Connexion USB avec RXD, TXD, DTR.

Pour la programmation du µC auxiliaire. Ne sert normalement que lors de la fabrication de la carte.

µC auxiliaire utilisé pour la programmation du µC principal.

USB ⇄ RXD, TXD, DTR.

1 seul implanté.

Jack alimentation + au milieu. 6,7V min.

Protection contre inv. polarités.

Connecteur SIL bord de carte.

Pour la prog. du µC principal sans passer par la liaison USB. Attention risque d’effacement du bootloader.

µC principal (en réalité ATmega 328P). Contient un bootloader pour l’autoprogrammation (utilisation RXD, TXD).

Signaux pour la programmation du µC principal


Pour RàZ du µC principal par le µC auxiliaire avec DTR.

Dropout 1V.

1 seul implanté.

Connecteur SIL bord de carte.
4.3 ALIMENTATION

4.3.1 SCHEMA

La carte Arduino peut être alimentée uniquement par le câble USB qui sert à la programmation.
Pas d’alimentation par le jack PWRIN (VIN = 0V initialement).

<table>
<thead>
<tr>
<th><strong>Fonctionnement</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Lors de la connexion de la fiche USB, diode de T1 conduit → ≈4,3V sur VCC (normalement 5V) → le régulateur 3,3V fonctionne et le comparateur U1A (ou IC3A selon implantation) est alimenté. 3,3V &gt; V CMP → sortie du comparateur à 0, Vs proche de 0V → VGS T1 (canal P) &lt; -4V → le transistor devient fortement conducteur RDS faible → VCC ≈ V USBVCC</td>
<td></td>
</tr>
<tr>
<td>VCC ≈ 5V</td>
<td></td>
</tr>
<tr>
<td>Valeur relevée VCC = 4,91V → V CMP ≈ 2,45 V &lt; 3,3V. La sortie du comparateur reste tj à 0.</td>
<td></td>
</tr>
</tbody>
</table>

Valeur de VIN
Si V PWRIN = 0, le régulateur IC2 (ou IC1 selon implantation) est utilisé à l’envers. Il conduit en sens inverse.
Valeurs relevées : VIN = 4,29V si aucune autre carte d’extension (shield) n’est connectée.
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

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

Connecteurs DIGITAL (PWM ~)

<table>
<thead>
<tr>
<th>Nom Arduino</th>
<th>Broche µC</th>
<th>Autres fonctions</th>
<th>Remarque</th>
</tr>
</thead>
<tbody>
<tr>
<td>RX ← 0</td>
<td>PD0</td>
<td>RXD / PCINT16</td>
<td>liaison série asynchrone. Aussi utilisée pour auto prog µC. R d’isolement de 1K pour cette fonction. Voir plus loin pour la programmation du µC</td>
</tr>
<tr>
<td>TX → 1</td>
<td>PD1</td>
<td>TXD/PCINT17</td>
<td>liaison série asynchrone. Aussi utilisée pour auto prog µC. R d’isolement de 1K pour cette fonction. Voir plus loin pour la programmation du µC</td>
</tr>
<tr>
<td>2</td>
<td>PD2</td>
<td>INTO/PCINT18</td>
<td></td>
</tr>
<tr>
<td>~3</td>
<td>PD3</td>
<td>INT1/OC2B/PCINT19</td>
<td>Peut être utilisée pour générer des signaux dont PWM.</td>
</tr>
<tr>
<td>4</td>
<td>PD4</td>
<td>T0/XCK/PCINT20</td>
<td></td>
</tr>
<tr>
<td>~5</td>
<td>PD5</td>
<td>T1/OC0B/PCINT21</td>
<td>Peut être utilisée pour générer des signaux dont PWM.</td>
</tr>
<tr>
<td>~6</td>
<td>PD6</td>
<td>AIN0/OC0A/PCINT22</td>
<td>Peut être utilisée pour générer des signaux dont PWM.</td>
</tr>
<tr>
<td>7</td>
<td>PD7</td>
<td>AIN1/PCINT23</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>PB0</td>
<td>ICP1/CLKO/PCINT0</td>
<td></td>
</tr>
<tr>
<td>~9</td>
<td>PB1</td>
<td>OC1A/PCINT1</td>
<td>Peut être utilisée pour générer des signaux dont PWM.</td>
</tr>
<tr>
<td>~10</td>
<td>PB2</td>
<td>/SS/OC1B/PCINT2</td>
<td>Peut être utilisée pour générer des signaux dont PWM.</td>
</tr>
<tr>
<td>Nom Arduino</td>
<td>Broche µC</td>
<td>Autres fonctions</td>
<td>Remarque</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Arduino</td>
<td>~11</td>
<td>MOSI/OC2A/PCINT3</td>
<td>Liaison série synchrone SPI. Peut être utilisée pour générer des signaux dont PWM. Est aussi utilisée pour la programmation ICSP.</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>MISO/PCINT4</td>
<td>Liaison série synchrone SPI. Est aussi utilisée pour la programmation ICSP.</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>SCK/PCINT5</td>
<td>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. Voir plus loin la partie sur la programmation.</td>
</tr>
</tbody>
</table>

⚠️ 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 µC 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 2ème 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 example.
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 µC 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=cHGDjGglNohOpKR
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 programmateur 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 programmateur 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/OutilsDeDeveloppement
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://itpedia.nyu.edu/wiki/Arduino_in_Eclipse](http://itpedia.nyu.edu/wiki/Arduino_in_Eclipse)
- [http://www.codeproject.com/Articles/110859/Arduino-Unleashed](http://www.codeproject.com/Articles/110859/Arduino-Unleashed)

### 6.3.2 VISUAL MICRO = ARDUINO FOR VISUAL STUDIO (MICROSOFT)


Pb : un lien est cassé pour l’inscription.


### 6.3.3 AVR STUDIO 5


### 6.4 FLOWCODE

FlowCode est un EDI non spécifique à Arduino.

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

### 6.5 LABVIEW


- 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 µC PRINCIPAL PUIS UTILISATION

7.1 GENERALITES SUR LA PROGRAMMATION ET L’UTILISATION

Le µC 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 µC 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’apparait pas avec des versions plus anciennes du bootloader.

7.2 PROGRAMMATION ICSP (In Circuit Serial Programming)


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

Le logiciel utilisé pour la programmation du µC 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 programmateur. Or c’est ici le µC principal lui-même qui se ‘fait passer’ pour un programmateur. Ainsi, il récupère les données du programme pour s’autoprogrammer.

Le µC 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 +

7.3.5 PROGRAMMATION DEPUIS FLOWCODE

FlowCode utilise AVRdude. Pour désigner le µC 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 µC 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 Downloader/UploadEr) 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 µC 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 µC 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.spinellicreations.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 (SHEilds)

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

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

<table>
<thead>
<tr>
<th>Fabricant</th>
<th>Site Web</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://shop.snootlab.com">http://shop.snootlab.com</a></td>
<td><a href="http://shop.snootlab.com/6-arduino">http://shop.snootlab.com/6-arduino</a></td>
<td>Grand choix de cartes dont une avec borniers. Commandes ly-</td>
</tr>
</tbody>
</table>
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 (+ BORNIERS)

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
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 BPs sont réunis à une seule entrée analogique. Si plusieurs BPs sont simultanément appuyés, seul le plus haut sur le schéma est pris en compte.

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

Compatibilité Arduino + Flowcode : oui.

8.3 AFFICHEUR LCD LIAISON I2C
Vendu en kit par snootlab.
Référence : KIT-01051
24,00 € TTC

L’utilisation l’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).

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 :

<table>
<thead>
<tr>
<th>PWMn</th>
<th>DIRn</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>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.</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>tension aux bornes du moteur est égale à ≈+VIN</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>tension aux bornes du moteur est égale à ≈-VIN</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>Nom carte</th>
<th>Nom Arduino</th>
<th>Broche µC</th>
<th>Autres fonctions</th>
<th>Remarque</th>
</tr>
</thead>
<tbody>
<tr>
<td>PWMA</td>
<td>~3</td>
<td>PD3</td>
<td>INT1/OC2B/PCINT19</td>
<td>PWM avec timer 2</td>
</tr>
<tr>
<td>DIRA</td>
<td>12</td>
<td>PB4</td>
<td>MOSI/OC2A/PCINT3</td>
<td></td>
</tr>
<tr>
<td>PWMB</td>
<td>~11</td>
<td>PB3</td>
<td>INT1/OC2B/PCINT19</td>
<td>PWM avec timer 2</td>
</tr>
<tr>
<td>DIRB</td>
<td>13</td>
<td>PB5</td>
<td>SCK/PCINT5</td>
<td></td>
</tr>
</tbody>
</table>


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

<table>
<thead>
<tr>
<th>Nom Modkit</th>
<th>Nom Arduino</th>
<th>Broche µC</th>
<th>Autres fonctions</th>
<th>Remarque</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIRA</td>
<td>2</td>
<td>PD2</td>
<td>INTO/PCINT18</td>
<td></td>
</tr>
<tr>
<td>PWMA</td>
<td>~3</td>
<td>PD3</td>
<td>INT1/OC2B/PCINT19</td>
<td>non compatible Flowcode</td>
</tr>
<tr>
<td>DIRB</td>
<td>4</td>
<td>PD4</td>
<td>TO/XCK/PCINT20</td>
<td></td>
</tr>
<tr>
<td>PWMB</td>
<td>~5</td>
<td>PD5</td>
<td>T1/OC0B/PCINT21</td>
<td>non compatible Flowcode</td>
</tr>
</tbody>
</table>
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 COTÉE**

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.


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 CRÉER 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)
- 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.

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 :

<table>
<thead>
<tr>
<th>Sortie carte Arduino</th>
<th>Broche du µC</th>
<th>Nom Flowcode</th>
<th>Entrée carte Ardumoto</th>
</tr>
</thead>
<tbody>
<tr>
<td>9 (broche 2 connecter IOH)</td>
<td>PB1 (OC1A)</td>
<td>PWM canal 1</td>
<td>PWMB (broche 11)</td>
</tr>
<tr>
<td>10 (broche 3 connecter IOH)</td>
<td>PB2 (OC1B)</td>
<td>PWM canal 2</td>
<td>PWMA (broche 3)</td>
</tr>
<tr>
<td>12</td>
<td>PB4</td>
<td>B4</td>
<td>DIRA (broche 12)</td>
</tr>
<tr>
<td>13</td>
<td>PB5</td>
<td>B5</td>
<td>DIRB (broche 13)</td>
</tr>
</tbody>
</table>

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 / Ardumoto 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 = Vref / $2^{10}$ = 4,88 mV

N = Ve/Vref/2

Capacité d’échantillonnage / blocage du CAN : 14 pF. Si le circuit d’attaque a une résistance équivalente de 100 KΩ (cas pour la voie 1) à Constante de temps = 1,4 µS. Pour obtenir une précision de 1%, il faut attendre au moins 5 à 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 beaucoup 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 le 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

<table>
<thead>
<tr>
<th>1 élément</th>
<th>Tension nominale (à mi charge environ)</th>
<th>Tension max pleine charge</th>
<th>Seuil pour autoriser le fonctionnement</th>
<th>Seuil min pour arrêter le fonctionnement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3,7V</td>
<td>4,2V</td>
<td>3,55V</td>
<td>3,3V</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>Tension</th>
<th>3,55V</th>
<th>3,3V</th>
<th>2,5V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Résultat CAN Avec Vréf = 5V</td>
<td>727</td>
<td>675</td>
<td>512</td>
</tr>
</tbody>
</table>

10.4.4 CONNECTEURS

Connecteurs à utiliser pour les batteries :

<table>
<thead>
<tr>
<th>Désignation</th>
<th>Ref constructeur</th>
<th>Ref Farnell</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embase 2 contacts Tyco type PE</td>
<td>AMP - 1586041-2 - EMBASE CI COUDEE 2 VOIES UL94V-2</td>
<td>811-5974</td>
</tr>
<tr>
<td>Boîtier femelle Tyco PE2</td>
<td>AMP - 794954-2 - BOITIER FEMELLE 2 VOIES 94V-2</td>
<td>811-6261</td>
</tr>
</tbody>
</table>

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 :

<table>
<thead>
<tr>
<th>LCD</th>
<th>RS</th>
<th>RD ou RW</th>
<th>EN</th>
<th>DB4</th>
<th>DB5</th>
<th>DB6</th>
<th>DB7</th>
</tr>
</thead>
<tbody>
<tr>
<td>N° broche LCD</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>Bit d’un Port µC</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Repères des broches sur le connecteur Arduino</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broche µC</td>
<td>PD0</td>
<td>PD1</td>
<td>PD2</td>
<td>PD3</td>
<td>PD4</td>
<td>PD5</td>
<td>PD6</td>
<td>PD7</td>
</tr>
<tr>
<td>LCD</td>
<td>RS</td>
<td>RW</td>
<td>EN</td>
<td>DB4</td>
<td>DB5</td>
<td>DB6</td>
<td>DB7</td>
<td></td>
</tr>
</tbody>
</table>

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

![Schema Structurel](image)

### 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 :
R2 = 2,2K  Valeur choisie arbitrairement dans la série E12.

BP3 appuyé, on veut VA0 ≈ 1V
1 = 5 * R3 / (R2 + R3) → R2 = 4 R3 → R3 = R2 /4 = 550Ω → valeur normalisée 560Ω

BP4 appuyé, on veut VA0 ≈ 2V
2 = 5 * (R3 +R4) / ((R2 + R3) + R4) → 2 R2- 3 R3 = 3 R4 → R4 = 2/3 R2 - R3 = 906 Ω → valeur normalisée 820Ω

BP5 appuyé, on veut VA0 ≈ 3V
3 = 5 * (R3 + R4 + R5) / ((R2 + R3 + R4) + R5) → 3 R2 - 2 R3 - 2 R4 = 2 R5 → R5 = 3/2 R2 - R3 - R4 = 1920 Ω → valeur normalisée 1800Ω

BP6 appuyé, on veut VA0 ≈ 4V
4 = 5 * (R3 + R4 + R5 + R6) / ((R2 + R3 + R4 + R5) + R6) → 4 R2 - R3- R4- R5 = R6 = 5620 → valeur normalisée 5,6 KΩ

<table>
<thead>
<tr>
<th>BP appuyé</th>
<th>Aucun</th>
<th>BP2</th>
<th>BP3</th>
<th>BP4</th>
<th>BP5</th>
<th>BP6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tension approximative en A0 (V)</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

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 chaine 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 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 :

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 :
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".
Caractéristiques (données par le constructeur) :
- 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, once 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.


13.2 PROJETS DISPONIBLES SUR INTERNET

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 :

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\build\xxxx.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 1ère 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 leanardo 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://openhardwareplatform.blogspot.fr/2011/03/inside-arduino-build-process.html
PROCESSING

interface graphique programmable côté PC pour le système Arduino