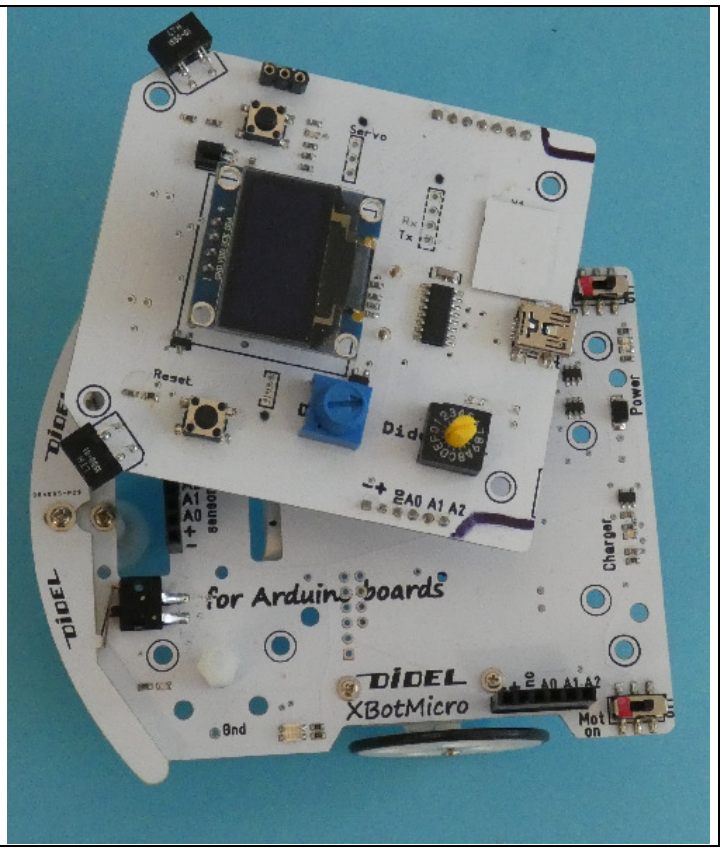
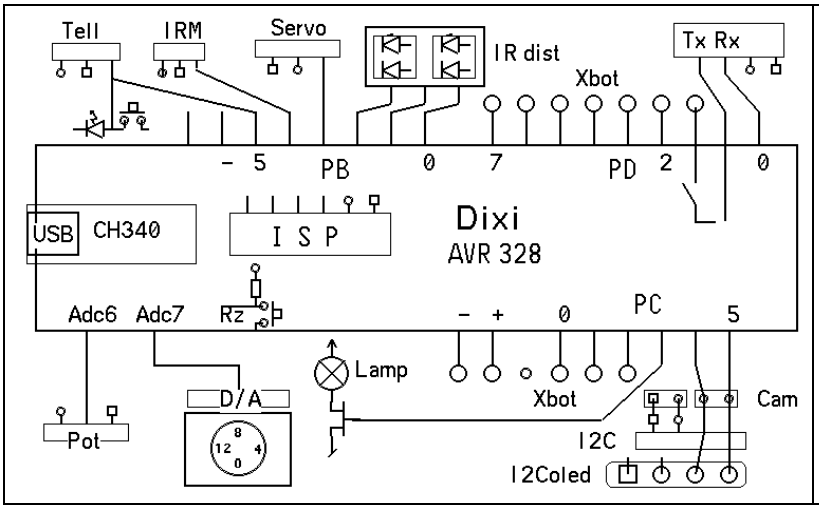
Dixi – carte "Arduino" pour la base Xbot

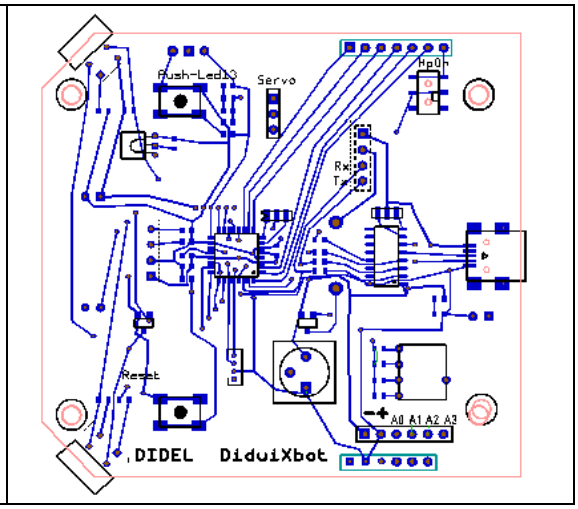
Dixi est une carte Arduino Uno/Duemilanove compatible qui s'enfiche sur la base Xbot et ajoute des fonctionnalités très intéressantes:

- Poussoir et Led sur la pin 13 comme pour le Diduino2
- Capteurs de distance pour éviter les murs (Dist2Ir)
- Capteur de télécommande infrarouge IRM
- Connecteur analogique en parallèle avec le connecteur avant du Xbot
- Lampe pour suivi de lumière
- Potentiomètre pour saisir une valeur
- Commutateur rotatif pour choisir 16 valeurs ou actions
- Connecteur pour Oled SSD1306
- Connecteur I2C pour ajouter des capteurs
- Connecteur Tell
- Connecteur série pour Bluetooth ou Wifi
- Connecteur pour un servo
- Interrupteur pour le buzzer du Xbot



Le DixiBot est une base Xbot avec la carte Dixi à califourchon.





Pin	Port		Pin	Port		Pin	Port	
0	PD0	Rx Conn	8	PB0	IrG	14	PC0	A0 base+con
1	PD1	**Tx Hp**	9	PB1	IrD	15	PC1	A1 base+con
2	PD2	ObsG	10	PB2	IrLed	16	PC2	A2 base+con A4
3	PD3	ObsD	11	PB3	Servo	17	PC3	con
4	PD4	RecG	12	PB4	IrModule	18	PC4	I2C SDA
5	PD5	AvG	13	PB5	Led/Pous/Tell	19	PC5	I2C SCL
6	PD6	AvD					An6	Pot
7	PD7	RecD					An7	

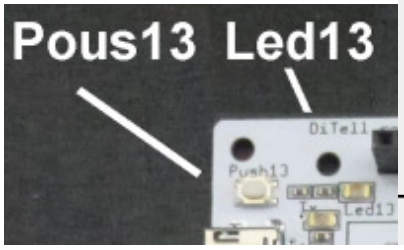
Signaux vers la base Xbot

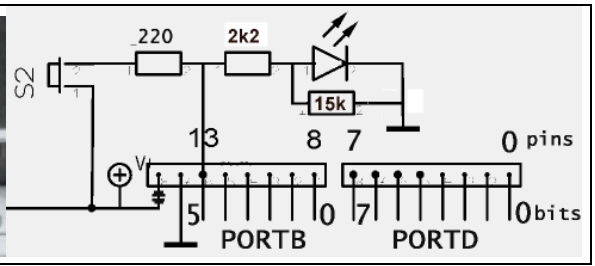
Definition file

Les fichiers de définition se trouvent dans les exemples de programmes sous **www.didel.com/DixiBot.zip**. Les documentations Xbot proposent des programmes compatibles, avec des notation souvent différentes.

Led et Poussoir

La Pin13 (PortB bit5) est traditionnellement réservée pour l'aide à la mise au point. Dixi a comme pour la carte Diduino un poussoir en parallèle.





Pédagogiquement, la combinaison permet de présenter et tester la Led, puis le poussoir, puis les contraintes imposées par le passage de l'utilisation de la Led au poussoir.

En mode Led la pin 13 est en sortie et une résistance de 220 Ohm neutralise l'effet du poussoir, la sortie est dominante. La Led s'allume si on programme l'état "1".
En mode poussoir, la pin 13 est en entrée, le poussoir envoie du courant pour allumer la Led, et on peut calculer la tension sur l'entrée dans les 2 modes.
Par défaut, on est en mode Led, Led allumée ou éteinte. Si on passe en mode "push" il faut attendre quelques microsecondes pour que le circuit se stabilise. On revient en mode Led dès que l'on a lu l'état. Documentation complémentaire sur www.didel.com/diduino/Poussoir.pdf
Anglais: www.didel.com/diduino/PushButton.pdf

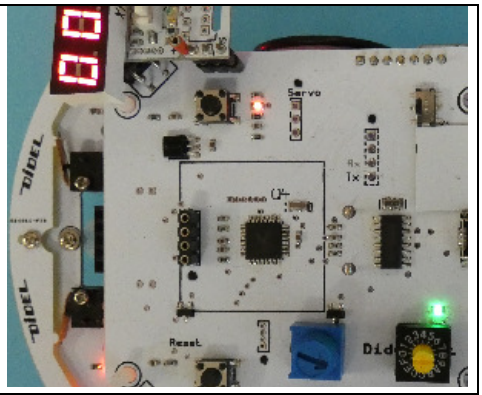
C	`#define bLP 5  // LED/Push bit on PORTB`
	`#define PushMode  bitSet (DDRB,bLP)`
	`#define LedMode  bitClear  (DDRB,bLP)`
	`#define  LEDOn  bitSet  (PORTB,bLP)`
	`#define  LEDOff LedMode; bitClear(PORTB,bLP)`
	`#define  PushOn PushMode; Del60; PINB&(1<<bLP); LedMode`
	`void SetupLed { bitSet (DDRB,bLed); }`

Arduino	`#define LP 13 // LED/Push pin`
	`#define PushMode  pinMode (LP,INPUT)`
	`#define LedMode  pinMode (LP,OUTPUT)`
	`#define  PushOn  PushMode; delayMicroseconds(60); digitalRead(LP);`
	`#define  LedOn  LedMode; digitalWrite (LP,HIGH)`
	`#define  LedOff  LedMode; digitalWrite (LP,LOW)`
	`void SetupLed { pinMode (Led,OUTPUT); }`

Tell sur pin 13

La pin 13 peut commander un affichage qui complète ou remplace le terminal série quand la carte est autonome. C'est plus rapide et souvent bien plus pratique que ce terminal. La fonction Tell(v16); envoie 16-bit data sur l'affichage Tell.
Tell a un poussoir pour afficher en hexadécimal ou décimal sans modifier le programme!
Voir www.didel.com/diduino/DiTell.pdf
Dixi a un pot en A6, voici un vraiment petit programme de test:

```
void setup() { } // initialization Arduino par défaut
void loop(){
   Tell (analogRead(A6)); delay (100);
}
```



Moteurs du Xbot

Le moteur est câblé de façon à pouvoir programmer en Arduino pour faire avancer le robot en PWM. Plusieurs librairies pour des shields moteur sont utilisables.
Exemple simple:

```
void setup() { } // initialization Arduino
```

Les primitives dans Xbot.h traitent chaque moteur séparément :
AvG; AvD; pour avancer
RecD; RecG; pour reculer
StopG; StopD; pour s'arrêter

```
byte PwmValue;
void loop(){
  digitalWrite (4,LOW);
  analogWite (5,pwmValue);
  analogWite (6,pwmValue);
  digitalWrite (4,LOW)
}
```

A noter que StopG,D mettent le moteur en roue libre, les deux sorties à zéro.
BlockG; BlockD; mettent les deux sorties à un, les leds rouges et vertes sont donc allumées, le moteur est court-circuité et se freine plus vite. Dans ce mode, on a plus de peine pour tourner le moteur à la main.

Programme de test

Une boucle qui agite le robot oblige de lui courir après pour l'arrêter. Testons tout de suit l'utilité du poussoir et la subtilité de la commutation Led/Poussoir.
Le programme attend que l'on presse, et clignote pour le signaler.
Si on presse, il y a une attente pour positionner le robot avant de démarrer les mouvements.
Notez la façon "fonctionelle" qui définit les différents délais.
Mettre plusieurs instructions par lignes n'est pas recommandé dans les programmes C complexes, mais ici c'est tellement plus clair!
Au reset, le moteur est libre. Après une exécution il est bloqué. Le moteur et l'ampli moteur ne sont pas assez bon pour que la différence soit bien marquée.

```
//TestMot.ino
// Wait for action on pushbutton 13 and move for few
seconds

#include "XbotDef.h"

void setup() {
  SetupXBot();
}
#define Pdel delayMicroseconds(50)   // Push delay
#define Bdel delay (150)   // blink half period
#define Sdel delay (500) // start delay
#define Mdel delay (500) // move delay
void loop() { // blink and wait for a push//
  while (!PushOn) {
  LedMode; LedOn; Bdel; LedOff; Bdel;
  PushMode; Pdel;
  }
  Sdel;  // Un moteur après l'autre
   AvG; Mdel; RecG; Mdel; StopG; Mdel;
   AvD; Mdel; RecD; Mdel; StopD; Mdel;
  Sdel;  // Avance recule avec stop intermédiaire
   AvG;  AvD;  Mdel; StopG; StopD; Mdel;
   RecG; RecD; Mdel; StopG; StopD; Mdel;
  Sdel;  // Avance recule sans stop intermédiaire
   AvG; AvD; Mdel;
   RecG; RecD; Mdel; StopG; StopD; Mdel;
  Sdel;  // Avance recule avec blocage
   AvG;  AvD;  Mdel; BlockG; BlockD; Mdel;
   RecG; RecD; Mdel; BlockG; BlockD; Mdel;
}
```

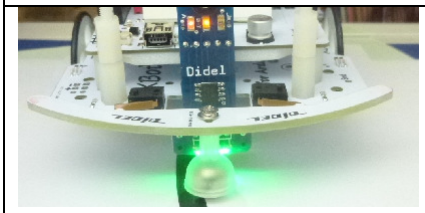
Delais

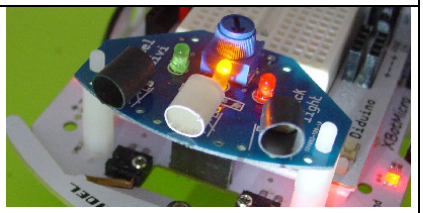
Nous définissons dans tous nos fichiers XxxDef.h deux fonctions de retard qui bloquent comme le delay (); fonction, mais ne lancez pas l'interruption et enregistrez 400 octets de code. Del (); est un retard de correction de 60 µS et DelMs (); est la même chose que delay (). C'est une fonction qui appelle deux boucles ne faisant rien. En tant que fonction, la première lettre est une majuscule.

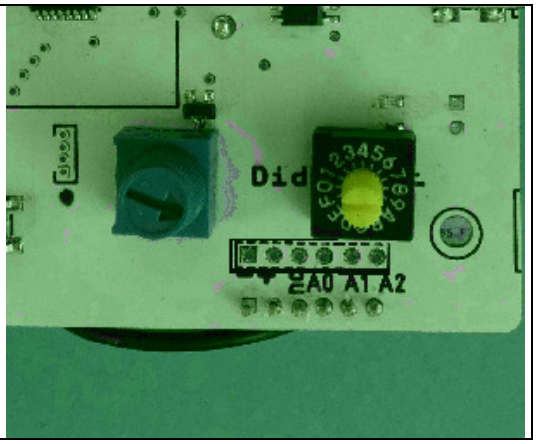
```
void Del() { // 50us no parameter
   volatile byte i=50;   // 50 30us
   while (i--) {}
}
void DelMs (int dm) {
   for (volatile int i=0; i<dm; i++) {
      for (volatile int j=0; j<800; j++) {}  // 16MHz
   }
}
```

Entrées A0 A1 A2 A4

Un connecteur à 6 reçoit le 3.7V du Lipo de la base Xbot et transmet le 5V pour charger ce LiPo. Les signaux A0, A1, A2 de la carte Dixi sont transmis au connecteur avant du Xbot, voir www.didel.com/xbot/Xsensors.pdf.
Ces signaux, complété par A3, sont disponibles avec le Gnd et Vcc sur un connecteur femelle.

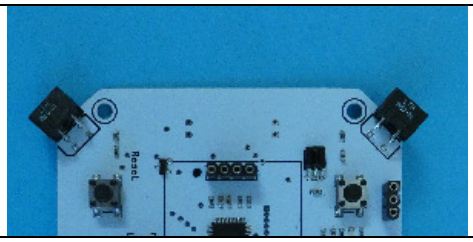






Capteurs de distance (Dist2Ir)

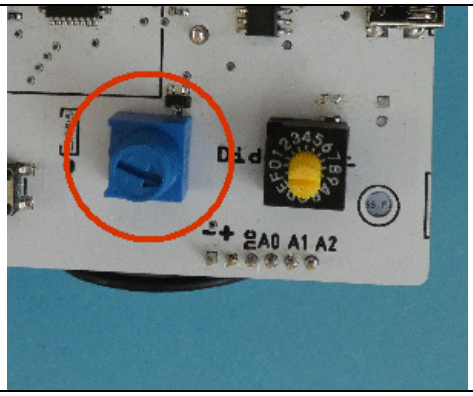
Le capteur de distance Dist2Ir est sur la carte, avec les mêmes spécifications.
https://www.didel.com/prof/Xdist2Ir.pdf



Potentiomètre et commutateur rotatif

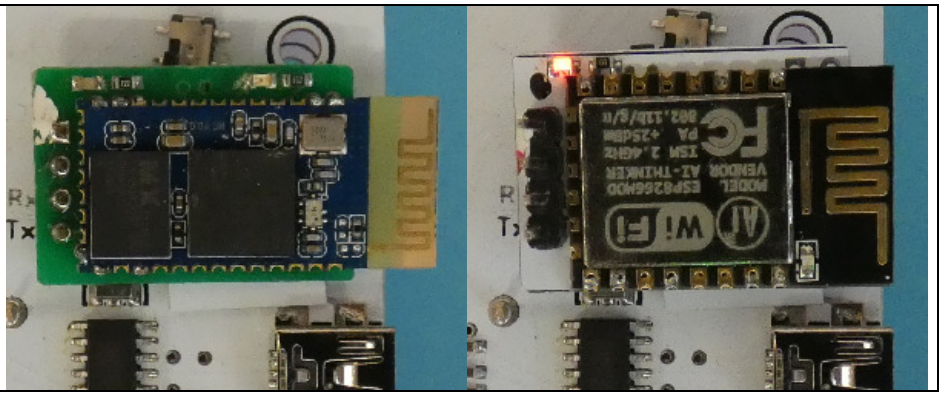
Un potentiomètre est câblé sur l'entrée AN6 de l'AVR 328; cette pin n'est pas accessible sur les cartes Arduino. Le potentiomètre se lit avec un analogRead(A6);

Le commutateur à 16 position utilise AN7 via un convertisseur D/A. Les 16 positions correspondent à 16 tension qu'une fonction assigne sur les 16 valeurs à utiliser dans un switch/case. https://www.didel.com/digrove/DgCod16



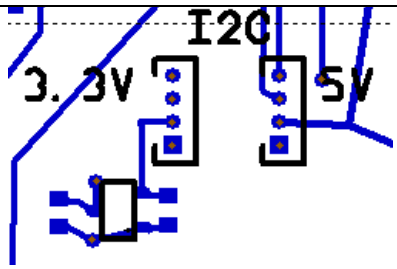
Connecteur série UART/Bt/Wifi

Les signaux Rx et Tx de l'AVR328, utilisés lors de la programmation via USB, son disponible sur un connecteur à 4 compatible avec des modules Bt et Wifi.
Tx est aussi envoyé via un interrupteur vers le haut-parleur de la base Xbot.



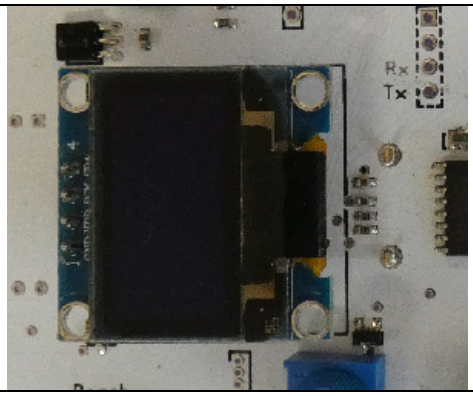
Connecteurs I2C

Les capteurs I2C existant sont annoncés 5V ou 3.3V. Les capteur 3-5V fonctionnent pendant le développement (5V) et en mode autonome (3.7V) sur un premier connecteur. Le capteurs 3.3V sont alimentés par un régulateur à partir du 5V ou 3.7V.

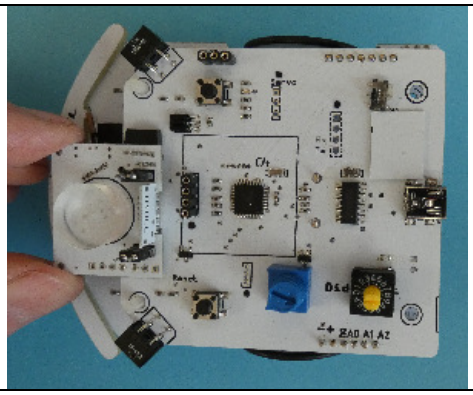


Connecteur I2C pour Oled SSD1306

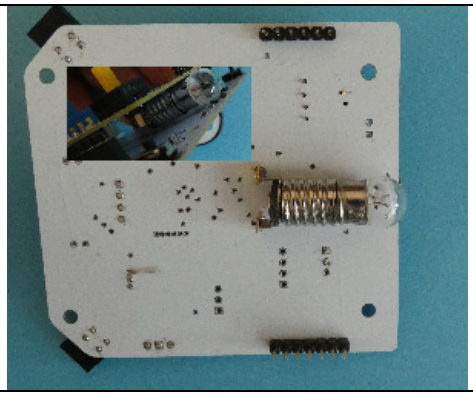
Un connecteur pour Oled.html 64x128, type SSD1306 est placé au centre de la carte. Le module de 32x128 est compatible, en ajoutant une instruction de configuration dans le set-up.



Connecteur I2C pour Caméra linéaire

La caméra linlaire en développement (96 pixels) sera un exclave I2C avec un Oled local pour afficher la courbe d'intensité lumineuse et des paramètres.	

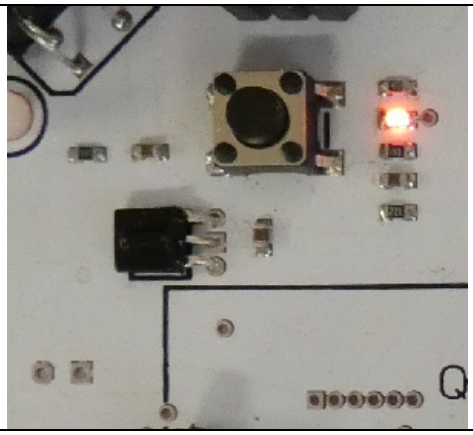
Lampe sous le PCB

Une application motivante pour un groupe qui a des Dixibots est de programmer des chenilles processionaires. Une ampoule a été prévue à l'arrière, pour avoir une lumière plus omnidirectionnelle qu'une Led. Il faut utiliser des ampoules 5V 50mA, et pas des ampoules de lampe de poche qui consomment trop.	

Connecteur servo ou RGB strip

Un connecteur avec alimentation et un signal connecté à la pin Arduino PB3 peut être utilisé pour lire un signal en tout-ou-rien, ou écrire une valeur sur une Led, un servo, un strip de leds RGB, etc..	

Récepteur IR (IRmodule)

Un récepteur IR compatible avec toutes les télécommandes ex lié à la pin PB4. Une solution simple compatible avec toutes les télécommandes est bien documentée : www.didel.com/TelecommandeIrSimple.pdf	

jdn rev 180909