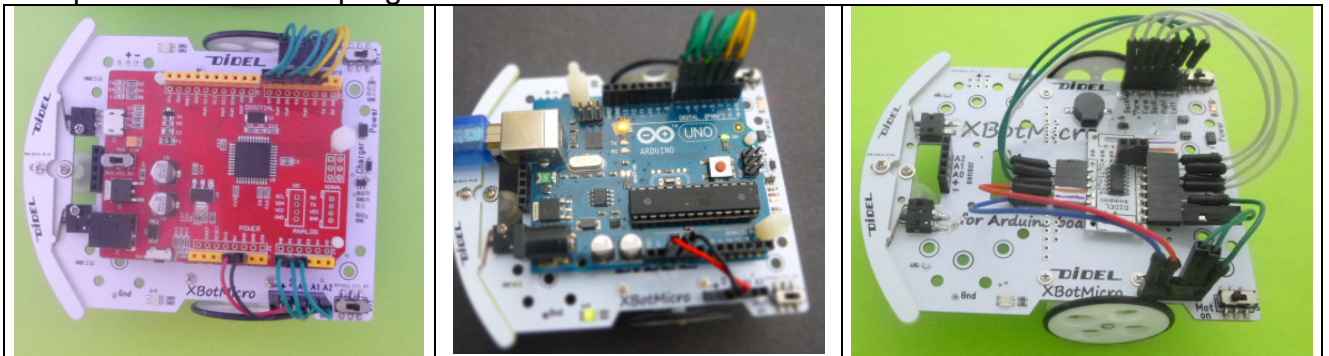




Programming the xBot On-off motor control and whiskers

You have now a working Xbot according to www.didel.com/robots/XBotMicroEn.pdf

The Xbot is controlled by 10 pins, with an arrangement that makes the wiring easy with an Arduino. Due to the availability of PWM and analogue inputs only on certain pins, the low level software depends on Arduino on these choices. But using macros and functions, all our C software will not depend on Arduino. You just need a better understanding of hardware and software to start with a non-Arduino microcontroller. Then you will be compatible with all our programs



Let us work at the functional level, and use definitions, macros and functions that hide the low lever hardware. The definition file Xbot.h for the Arduino Uno/Duemilanove is found in appendix.

```
#define ForwL bitSet (PORTD,bForwL); bitClear(PORTD,bBackL)
```

On/off motion control

Left motor is controlled by 2 pins. The macro ForwL; acivate these pins in such a way the motor spins in the good direction. This is declared in the XbotE.h file and assume the board is wired as documented. For instance:

```
#define bForwL 5
#define ForwL bitSet (PORTD,bForwL); bitClear(PORTD,bBackL)
```

If you use different microcontrollers, you should name that definition file more explicitly: XbotUno.h, XbotTiny24, XbotPinguino. The rest of your C-programs will be compatible.

Motor control	
ForwL;	ForwR;
BackL;	BackR;
StopL;	StopR;
Stop;	
Whiskers	
ObsL	ObsR
Loudspeaker and arduino led	
SpOn;	LedOn;
SpOff;	LedOff;
SpToggle;	LedToggle;
Examples:	
ForwL; ForwR;	
delay (1000); Stop;	
Move for 1 second	
if (ObsR ObsL) {	
SpOn; delayMicrosecond (500);	
SpOff; delayMicrosecond (500);	
}	
Beep if against an bstacle	

If you are not familiar with #defines and reference files, see www.didel.com/coursera/LC.pdf (in french)

and www.didel.com/diduino/ArduinoInclude.pdf

All the programs can be found under www.didel.com/xbot/XbotBegin.zip

Program 1 – Avoid obstacles

When the robot touch an object or the walls, he has to back and turn significantly.

```
// AvoidObstacle1.ino
// Back and turn a little when an obstacle is found.
#include "XBotE.h"

void setup() { // initialisation
  SetupXbotE();
}
#define DelRef 300 // back delay
#define DelRot 200 // rotate delay
void loop() {
  ForwR; ForwL;
  if (ObsR) { // back and turn
    BackR; BackL; delay(DelRef) ;
    StopR; delay(DelRot);
    StopL;
  }
  if (ObsL) { //on recule et tourne
    BackR; BackL; delay (DelRef);
    StopL; delay(DelRot);
    StopR;
  }
}
}
```

Do not forget to save, and save under a new name when you change the fonctionnality.

See www.didel.com/diduino/ArduinoInclude.pdf



You can use Arduino functions in the definition, It makes the program longer and slower, but this is not a critical point. What is important is the program must refer to any robot with 2 motors and 2 whiskers.

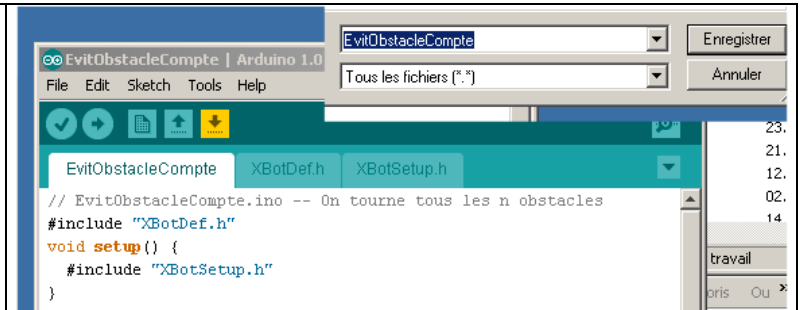
Mais nous avons fait une grave erreur en disant dans le programme qu'il recule de 0.3 secondes. C'est pas valable pour un autre robot! Il aurait fallu écrire DelReculer et déclarer au début du programmes, avec les déclarations matérielles

```
#define DelReculer 300
```

Program 2 – Avoid blocking

Depending on the obstacle, the robot may enter an endless movement. One trick is to count the obstacles, and every 5, for instance make a special movement.

When you modify the program or an #include, it is not saved on the disk. Compilation is done with what is seen on the screen. You have to save with CTRJ-s or the yellow icon next picture. .



```
...
byte countObst = 0 ;
#define nbObst 5
#define DelTouComptObst 800
void loop() {
  ForwRight; ForwLeft;
  ForwR; ForwL;
  if (ObsR) { // back and turn
    BackR; BackL; delay(DelRef);
    StopR; delay(DelRot);
    StopL;
  }
  if (ObsL) { //on recule et tourne
    BackR; BackL; delay (DelRef);
    StopL; delay(DelRot);
    StopR;
  }
  if ( compteObst > nbObst ) {
    compteObst = 0 ;
    TourneD;
    delay (DelTouComptObst);
  } // end if
} // end loop
```

Program execution will be more easy to understand if functions are defined for the frequently executed tasks. selecting good names for functions is important.

```
void AvoidObsR () { //back and turn
  if (ObsR) { // back and turn
    BackR; BackL; delay(DelRef);
    StopR; delay(DelRot);
    StopL;
  }
}
```

```
void AvoidObsL () { //back and turn
  if (ObsL) {
    BackR; BackL; delay (DelRef);
    StopL; delay(DelRot);
    StopR;
  }
}
```

Note these functions are not blocking. Rewrite the previous program, using them.

Program 3 – Follow a wall

Following a wall is interesting, one need to adapt the robot behaviour to the shape of the Arena. The robot has to search for the wall, doing small turns in its direction. Inside a square box, one can take care of the corners, with both whiskers being activated.

Naive functions that can be defined are

```
WhileNoWallMoveRight();
```

```
AvoidWallRight ()
```

```

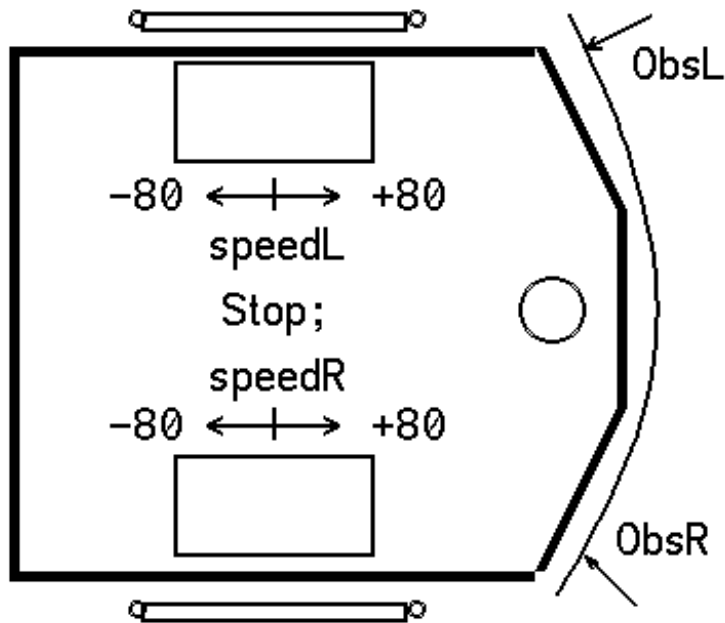
#define DelF 50 //
#define DelT 20 //
void WhileNoWallMoveRight() {
    ForwR; ForwL; delay(DelF);
    ForwR; StopL; delay(DelT);
    if (ObsR) { // back and turn
        BackR; BackL; delay(DelBack);
        StopR; delay(DelRot);
        StopL;
    }
}
void AvoidWallRight () {
}

```

Idées de programmes

Dessiner un polygone
 Suivre un mur
 Contourner l'extrémité d'une paroi
 Modifier la vitesse
 Corriger pour une ligne droite
 Tracer une spirale

Programming the xBot part 2 Speed control



Aller-retour

Programming the xBot part 3 State machines and interrupt

Selecting a demo
 Improving the

Programming the xBot part 4 Uson distance sensor

www.didel.com/xbot/DistSonar.pdf *DidtUson*

Programming the xBot part 5 Uson distance sensor

www.didel.com/xbot/Piste.pdf

Programming the xBot part 6 Uson distance sensor

www.didel.com/xbot/Suivi.pdf

Programming the xBot part 7 Uson distance sensor

www.didel.com/xbot/xDist2lr.pdf

Programming the xBot part 8 Appendix – reference files

<p>file Xbot.h defines several macros actions</p> <pre>ForwL; BackL; StopL; ForwR; BackR; StopR; Stop; ObsL; ObsR; SpOn; SpOff; SpToggle; LedOn; LedOff; LedToggle;</pre>	<p>The diagram illustrates the hardware connections for the xBotMicro for Arduino. It shows a top header with pins 1 through 13. Pin 1 is labeled 'bSpeaker' and is connected to a speaker. Pins 2 and 3 are labeled 'bWhiskR' and 'bWhiskL' respectively, connected to whisker sensors. Pins 4 and 5 are labeled 'bBackR' and 'bForwR', connected to the right motor (MotRight). Pins 6 and 7 are labeled 'bForwL' and 'bBackL', connected to the left motor (MotLeft). Pin 8 is connected to an LED. Pin 9 is labeled 'Sensor option' and is connected to a sensor module. The sensor module has a 'Gnd' and '+5V' connection. The xBotMicro for Arduino is shown with a 'Sensor option' header and a 'Didel' header. The Didel header has pins for '+', '-', 'Vin!', and 'A0 A1 A2'. The sensor module is connected to the Didel header.</p>
--	---

```

// XbotE.h
#include <Arduino.h>
#define bLed 5 // bit 5 PORTB (pin 13) // debug and DgTell
#define LedOn bitSet (PORTB,bLed)
#define LedOff bitClear (PORTB,bLed)
#define LedToggle PORTB ^= (1<<bLed)
#define PushMode bitClear (DDRB,bLed); PORTB=0;
#define LedMode bitSet (DDRB,bLed)
#define PushOn (PINB & (1<<bLed))

#define TstOn bitSet (PORTB,4) // debugging pin 12
#define TstOff bitClear (PORTB,4)
#define TstToggle PORTB ^= 1<<4
#define TstOut bitSet(DDRB,4)

//Definitions for motor bits
#define bBackL 4 // left motor on PORTD
#define bForwL 5 // pwm possible
#define bForwR 6 // pwm possible
#define bBackR 7 // right motor
// Definitions for the PFM
#define ForwL bitSet (PORTD,bForwL); bitClear(PORTD,bBackL)
#define BackL bitClear(PORTD,bForwL); bitSet (PORTD,bBackL)
#define StopL bitClear(PORTD,bForwL); bitClear(PORTD,bBackL)
#define ForwR bitSet (PORTD,bForwR); bitClear(PORTD,bBackR)
#define BackR bitClear(PORTD,bForwR); bitSet (PORTD,bBackR)
#define StopR bitClear(PORTD,bForwR); bitClear(PORTD,bBackR)
// Panic motor stop
#define StopAll PORTD &= 0b00001111
// a ajouter quand les variables seront definies: (?)
// pwmG=0; pwmD=0; pfmG=0; pfmD=0; vitG=0; vitD=0

// Whiskers on PORTD Active Low, can be written to activate the Leds
// but do not forget to update direction
#define bWhiskL 2 // PORTD,2)
#define bWhiskR 3
#define ObsL !(PIND&(1<<bWhiskL))
#define ObsR !(PIND&(1<<bWhiskR))

#define LedWhiskLOn bitSet (DDRD,bWhiskL);bitClear (PORTB,bWhiskL)
#define LedWhiskLOff bitClear (DDRD,bWhiskL)
#define LedWhiskROn bitSet (DDRD,bWhiskR);bitClear (PORTB,bWhiskR)
#define LedWhiskROff bitClear (DDRD,bWhiskR)

// Loudspeaker on RxD pin1 powered through a capacitor, no static current
#define bSp 1 // TxD bit 1 PORTD
#define SpOn bitSet (PORTD,bSp)
#define SpOff bitClear (PORTD,bSp)
#define SpToggle PORTD ^= (1<<bSp)

void SetupXbotE() {
  DDRD |= 0b11110010 ; // motors and HP
  DDRD &= 0b11110001 ; // whiskers
  DDRB |= 0b11110000 ; // Led13 and Tst out
  DDRB &= 0b11110000 ; // encoders
  DDRC = 0b00000000 ; // all inputs I2C , sensors
}

```

jdn 140620/140719