

A close-up photograph of a robotic hand, possibly a Shadow Hand, holding a small, glowing yellow sphere. The scene is dramatically lit with red light, creating a high-contrast, futuristic atmosphere. The background is dark, and the red light reflects off the metallic surfaces of the hand and the sphere.

GRUPOS DE TRABAJO 2015-16

***“PROTOTIPOS ROBÓTICOS
PARA AS NOVAS OPTATIVAS LOMCE”***

CÓDIGO V1505066/7

IES Alexandre Bóveda (Vigo)

Control Robótico compatible con ARDUINO



POR QUÉ FACER UN CONTROL ELECTRÓNICO PARA O ROBOT?

Na parte mecánica do robot ...

- Deseñamos e construímos as pezas coa impresora 3D
- A montaxe facémola nós (profes ou alumnos)

Na parte electrónica do robot...

- Hai que escoller e mercar unha placa Arduino comercial
- Os prezos varían tamén moito, poden ser moi baratas ou moi caras
- Hai que montar un "puzzle" de placas cableadas: arduino+teclado+motores

Se facemos tamén o deseño e montaxe da parte electrónica...

- Pode ter todos os elementos necesarios para o robot e ser máis compacta
- Pode ser máis económica que se a mercamos (ou non, pero aprendemos máis = I+D)
- A placa de circuíto impreso pode facerse no taller ou encargarse e logo montar os compoñentes
- Tería que ser compatible co contorno de programación de Arduino

É POSIBLE? FÁCIL? BARATO? BONITO (iso seguro...)?

UNHA OLLADA Ó INTERIOR DE ARDUINO

Para responder ó anterior, primeiro estudiaremos as orixes e características de Arduino:

CARACTERÍSTICAS:

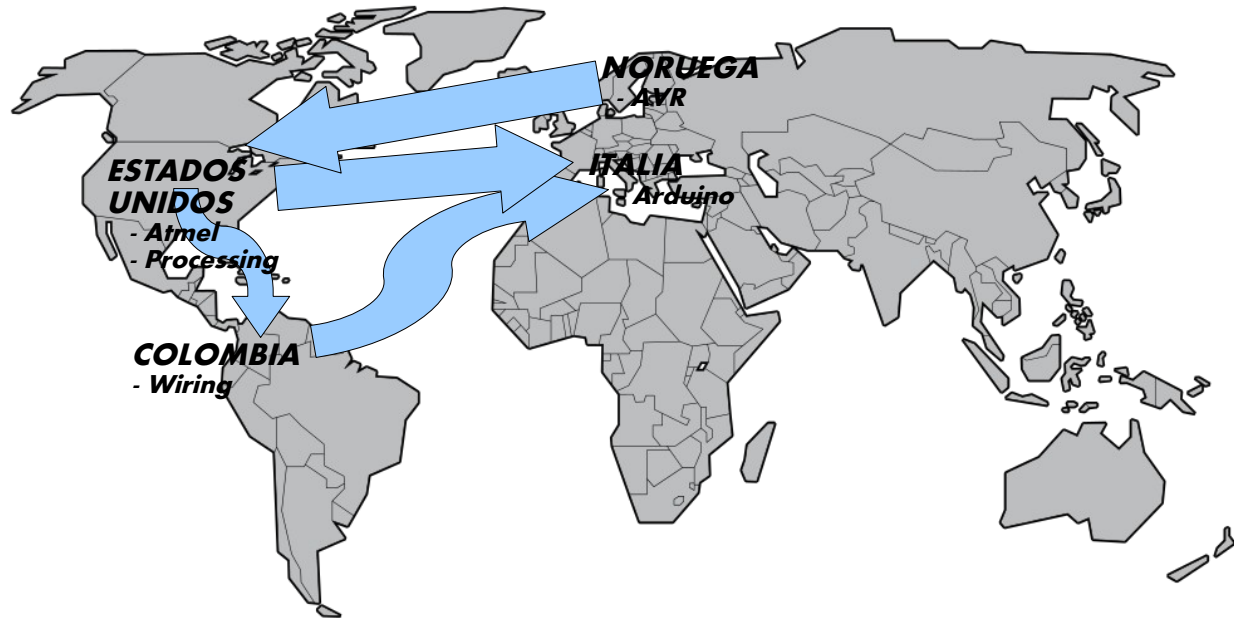
1) CONTORNO DE PROGRAMACIÓN

- Contorno de programación sinxelo
- Exemplos de programas e recursos (bibliotecas comunicacións, motores...)
- Comunidade desenvolvedores, foros, moita información
- Conexión co microcontrolador cun cable usb ou adaptador, non necesita un programador (caro).
- Transferencia de código e transmisión de datos por usb
- Programas de uso libre

2) HARDWARE (MICROCONTROLADOR)

- En hardware Arduino non aporta unha novidade importante
- Usa as características dos microcontroladores Atmel con arquitectura AVR
- Placas e esquemas de uso libre, hai moitos sistemas compatibles

ORIXES DE ARDUINO: ISTO SÍ QUE É GLOBALIZACIÓN



MICROCONTROLADORES ATMEL-AVR

Arquitectura AVR (https://en.wikipedia.org/wiki/Atmel_AVR): microcontroladores RISC (instruccións simples e alta velocidade) deseñados por dous estudantes (Alf-Egil Bogen e Vegard Wollan) no Norwegian Institute of Technology (Noruega) en 1996.

O primeiro circuito fíxose na empresa Nordic VLSI, que vende o deseño a Atmel (EEUU), aparecendo en 1997 a familia de microcontroladores AVR, dos cales forma parte a familia ATMEGA que usa Arduino. Os dous inventores seguiron desenvolvendo a arquitectura desde Noruega para esta nova empresa.

A nova arquitectura tivo un grande éxito, En 2003 levaban vendidos máis de 500 millóns de unidades.

Inicialmente eran compatibles en patillaxe con outra familia, a 8051 de Intel (AT90S8515). Na actualidade Atmel segue a fabricar ambas familias.

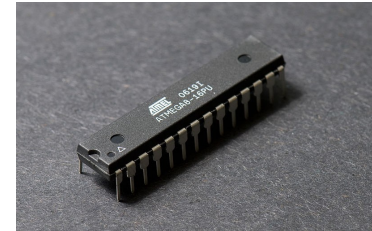
Estaban pensados para a programación en linguaxe C.

Aportaban novidades importantes, como usar memoria de programa tipo Flash (en lugar de ROM/EEPROM) ou execución de instrucións nun só ciclo de oscilador o que os facía máis rápidos que outros modelos da época (que empregaban 12 ciclos ou máis).

Levan buses de comunicación serie, memoria de datos integrada e outros recursos.

A memoria de programa ten prevista unha sección de código protexido que pode empregarse para gardar un programa de arranque, como se fora un pequeno sistema operativo.

Atmel crea tamén un contorno de aplicacións (IDE) chamado AVR Studio (actualmente Atmel Studio).

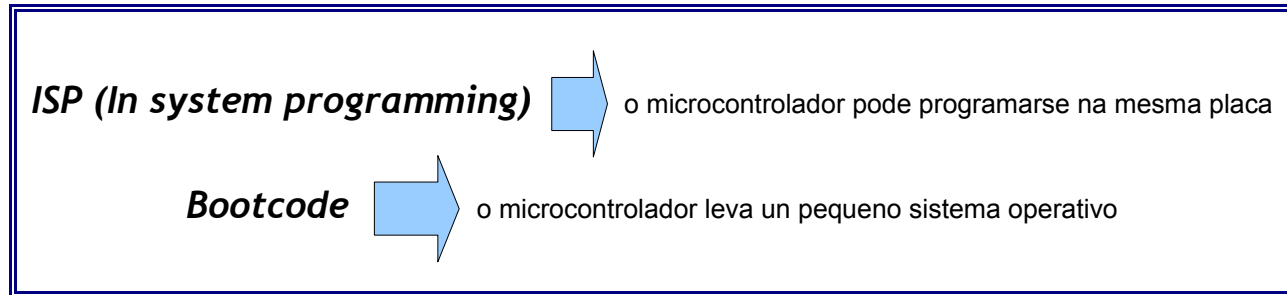


CARACTERÍSTICAS ARQUITECTURA AVR

- Multifunction, bi-directional general-purpose I/O ports with configurable, built-in [pull-up resistors](#)
- Multiple internal oscillators, including RC oscillator without external parts
- Internal, self-programmable instruction [flash memory](#) up to 256 kB (384 kB on XMega)
 - [In-system programmable](#) using serial/parallel low-voltage proprietary interfaces or [JTAG](#)
 - Optional boot code section with independent lock bits for protection
- On-chip debugging (OCD) support through JTAG or [debugWIRE](#) on most devices.
- Internal data [EEPROM](#) up to 4 kB
- Internal [SRAM](#) up to 16 kB (32 kB on XMega)
- External 64 kB little endian data space on certain models, including the Mega8515 and Mega162.
- 8-bit and 16-bit timers
 - [PWM](#) output (some devices have an enhanced PWM peripheral which includes a dead-time generator)
 - [Input capture](#) that record a time stamp triggered by a signal edge
- Analog comparator
- 10 or 12-bit [A/D converters](#), with multiplex of up to 16 channels
- 12-bit [D/A converters](#)
- A variety of serial interfaces, including [I²C](#), [RS-232](#), [RS-485](#), [Serial Peripheral Interface Bus](#)
- [Brownout](#) detection
- [Watchdog timer](#) (WDT)
- Multiple power-saving sleep modes
- Lighting and motor control ([PWM](#)-specific) controller models
- [CAN](#) controller support
- [USB](#) controller support
- [Ethernet](#) controller support
- [LCD](#) controller support
- Low-voltage devices operating down to 1.8 V (to 0.7 V for parts with built-in DC–DC upconverter)
- [DMA](#) controllers and "event system" peripheral communication.

CARACTERÍSTICAS ARQUITECTURA AVR

De todo o anterior, o máis importante...



O hardware Arduino está baseado nestas dúas características, engadindo un interfaz USB e un programa cargador propio ("bootloader").

Outras características dos micros AVR-mega que usa Arduino:

Portos I/O = entradas/saídas dixitais

Conversores A/D = entradas analóxicas (sensores)

PWM = control motores e potencia, saídas analóxicas

Portos serie (SPI, I2C) = comunicación por usb, bluetooth, etc

CONTORNO DE PROGRAMACIÓN: PROCESSING

Processing: contorno programación sinxelo baseado na linguaxe C.

Creado en 2001 por Benjamin Fry e Casey Reas cando estaban no Aesthetics and Computation Group at the MIT Media Lab.

Feito en Java e multiplataforma (Linux, Windows, Mac).

Estaba pensado para facer esquemas, diagramas, visualización de datos, arte electrónico (non para microcontroladores).

O desenvolvemento de Processing segue, en 2012 crearon a "Processing Foundation" con Daniel Shiffman, que se converte no terceiro coautor do proxecto.

https://en.wikipedia.org/wiki/Processing_%28programming_language%29
<https://processing.org/>

CONTORNO DE PROGRAMACIÓN: PROCESSING

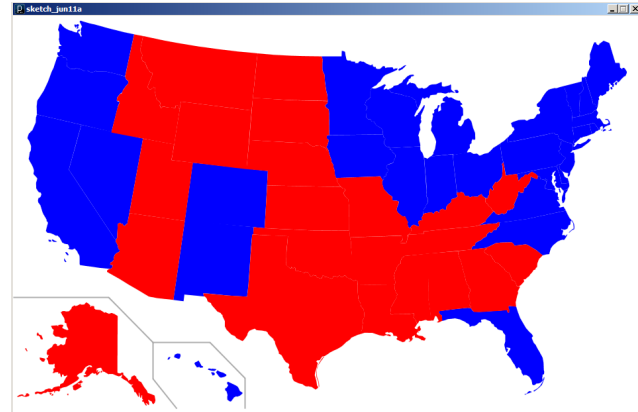
Exemplo de programa en Processing que debuxa un mapa:

```
PShape usa;
PShape state;
String [] Obama = { "HI", "RI", "CT", "MA", "ME", "NH", "VT", "NY", "NJ",
  "FL", "NC", "OH", "IN", "IA", "CO", "NV", "PA", "DE", "MD", "MI",
  "WA", "CA", "OR", "IL", "MN", "WI", "DC", "NM", "VA" };
String [] McCain = { "AK", "GA", "AL", "TN", "WV", "KY", "SC", "WY", "MT",
  "ID", "TX", "AZ", "UT", "ND", "SD", "NE", "MS", "MO", "AR", "OK",
  "KS", "LA" };

void setup() {
  size(950, 600);
  // The file Blank_US_Map.svg can be found at Wikimedia Commons
  usa = loadShape("http://upload.wikimedia.org/wikipedia/commons/3/32/Blank_US_Map.svg");
  smooth(); // Improves the drawing quality of the SVG
  noLoop();
}

void draw() {
  background(255);
  // Draw the full map
  shape(usa, 0, 0);
  // Blue denotes states won by Obama
  statesColoring(Obama, color(0, 0, 255));
  // Red denotes states won by McCain
  statesColoring(McCain, color(255, 0, 0));
  // Save the map as image
  saveFrame("map output.png");
}

void statesColoring(String[] states, int c){
  for (int i = 0; i < states.length; ++i) {
    PShape state = usa.getChild(states[i]);
    // Disable the colors found in the SVG file
    state.disableStyle();
    // Set our own coloring
    fill(c);
    noStroke();
    // Draw a single state
    shape(state, 0, 0);
  }
}
```



WIRING: CONTORNO DE PROGRAMACIÓN + PLACA MICROCONTROLADOR

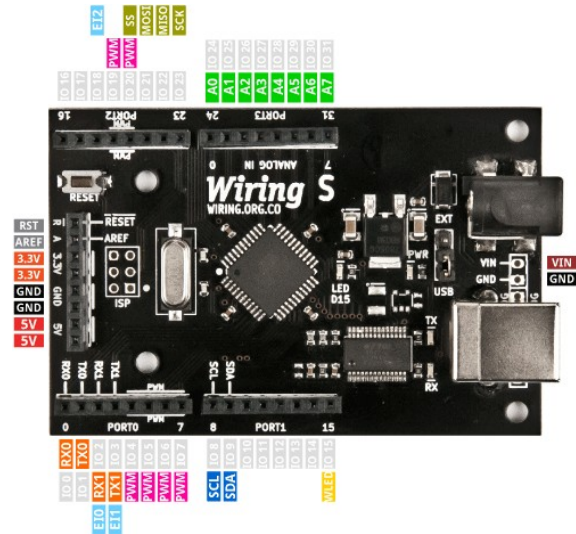
"Wiring" é unha plataforma de desenvolvemento de aplicacións electrónicas formada por unha linguaxe de programación, un contorno de desarrollo e una placa con microcontrolador.

Esta plataforma foi ideada polo colombiano Hernando Barragán en 2003 como tese doutoral no Interaction Design Institute en Ivrea, Italia. Actualmente o proxecto continúa na Escola de Arquitectura e Deseño na Universidad de Los Andes en Bogotá, Colombia.

Está baseado en Processing como contorno de programación, ó que se lle engade un conxunto de bibliotecas de funcións que facilitan o traballo con microcontroladores como *digitalRead()* ou *analogWrite()*.

A biblioteca de funcións "Wiring" segue a usarse en Arduino.

É un sistema aberto, tanto en hardware como en software, con Licenza Creative Commons.



Exemplo de placa de desenvolvemento actual de Wiring

https://en.wikipedia.org/wiki/Wiring_%28development_platform%29

<http://wiring.org.co/>

A PLATAFORMA ARDUINO

Arduino é un sistema de desenvolvemento de aplicacións electrónicas creado a partir de 2004 no Interaction Design Institute (Ivrea, Italia) a partir de Processing-Wiring, usando microcontroladores AVR-Atmega de Atmel.

O equipo resultante estaba pensado para os estudantes, era moi fácil de usar e tiña un custo moi reducido, polo que acadou rapidamente un grande éxito,

Os autores iniciais foron Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, e David Mellis.

Arduino ten o seu propio contorno de desenvolvemento (IDE) feito en Java, polo que tamén é multiplataforma.

A linguaxe de programación está baseada en Processing e usa a biblioteca de funcións do proxecto Wiring, e está pensado para introducir na programación a artistas ou persoas alleas á electrónica e informática.

Ten un editor de código moi sinxelo de usar e os programas se compilan e transfírense ó microcontrolador cun par de "clicks".

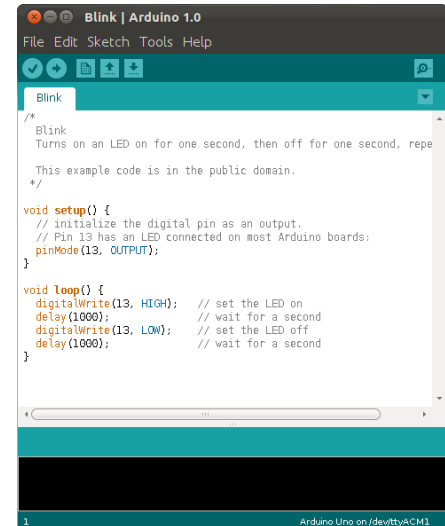
Referencias:

<https://en.wikipedia.org/wiki/Arduino>

<https://www.arduino.cc/>

Contribucións ó proxecto:

<https://www.arduino.cc/en/Main/Credits>

A screenshot of the Arduino IDE interface. The title bar reads "Blink | Arduino 1.0". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for opening files, saving, and running. The main text area displays the "Blink" sketch, which is a simple program to turn an LED on and off. The code is as follows:

```
/*  
 * Blink  
 * Turns on an LED on for one second, then off for one second, repe  
 *  
 * This example code is in the public domain.  
 */  
  
void setup() {  
  // initialize the digital pin as an output.  
  // Pin 13 has an LED connected on most Arduino boards:  
  pinMode(13, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(13, HIGH); // set the LED on  
  delay(1000);             // wait for a second  
  digitalWrite(13, LOW);  // set the LED off  
  delay(1000);             // wait for a second  
}
```

The status bar at the bottom indicates "1" and "Arduino Uno on /dev/ttyACM1".

A PLATAFORMA ARDUINO

Un programa feito en Arduino chámase “sketch” e os arquivos teñen a extensión “.ino”.

O programa típico Arduino ten dúas funcións:

- “setup()”, esta función contén código que se executa unha única vez ó principio do programa e se emprega para realizar operacións de inicio, establecer valores de datos, etc.
- “loop()”, esta función se repite indefinidamente ata que se desconecta a placa. Serve para revisar o estado de entradas e sensores e facer as operacións necesarias segundo o resultado (por exemplo, activar un relé, motor, led,...)

Exemplo de sketch Arduino (blink.ino)

```
#define LED_PIN 13

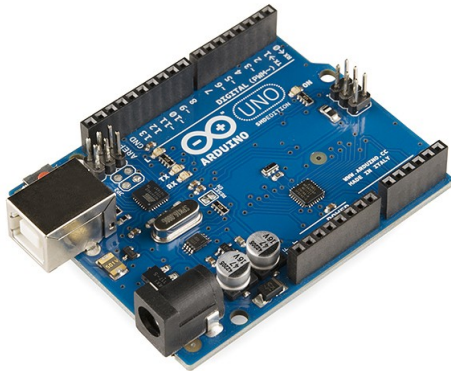
void setup() {
    pinMode(LED_PIN, OUTPUT);    // Enable pin 13 for digital output
}

void loop() {
    digitalWrite(LED_PIN, HIGH); // Turn on the LED
    delay(1000);                 // Wait one second (1000 milliseconds)
    digitalWrite(LED_PIN, LOW);  // Turn off the LED
    delay(1000);                 // Wait one second
}
```

A PLATAFORMA ARDUINO

Os sketches de Arduino se traducen mediante programas GNU (como o compilador AVR-GCC) que xa están incluídos no contorno de desenvolvemento. Tamén se emprega un programa de comunicacións de uso libre chamado avrdude (<http://savannah.nongnu.org/projects/avrdude>) para enviar o código resultante ó microcontrolador.

A placa de control ten un pequeno programa chamado Bootloader que tamén é parte de Arduino e se comunica co ordenador para organizar a transferencia de código. Gracias a isto non se precisan elementos externos como programadores e a tarefa de programación e depuración é rápida e sinxela.



Placa de control Arduino UnoR3



Programador AVRISP mkII

RESPONDENDO Á PREGUNTA INICIAL: Pode facerse un control compatible con Arduino?

Sí, unha placa compatible con Arduino pode facerse cun microcontrolador e uns poucos elementos máis (cuarzo, condensadores, resistencia).

Na documentación de Arduino hai un tutorial que explica cómo: "Building an Arduino on a breadboard", <https://www.arduino.cc/en/Main/Standalone>

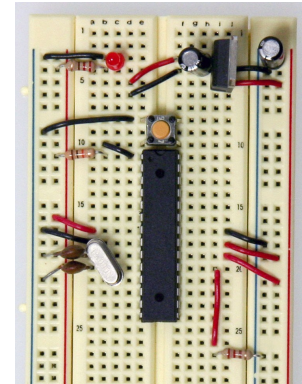
A NOSA PLACA: CRITERIOS DE DESEÑO

1) O elemento principal de Arduino é o microcontrolador. Usaremos o mesmo que levan moitas das placas Arduino (duemilanove, diecimila), o **Atmega328P**. Para que sexa fácil de montar e se poidan quitar escollemos a versión Atmega328P-PU (encapsulado paralelo pdip, 28 patas, separación 0,1"=2,54mm). Ollo, a patillaxe non é igual que na versión smd!!.

2) As placas Arduino levan o programa de inicio ("**Bootloader**"). Os micros comprados non o traen, polo que hai que gravalo. Veremos cómo facelo con outro tutorial: "From Arduino to a Microcontroller on a Breadboard", <https://www.arduino.cc/en/Tutorial/ArduinoToBreadboard>

A gravación pode facerse de varias formas: cun programador tipo AVRISP, co módulo hardware ArduinoISP, ou con outra placa Arduino cargando o sketch "Arduino as ISP".

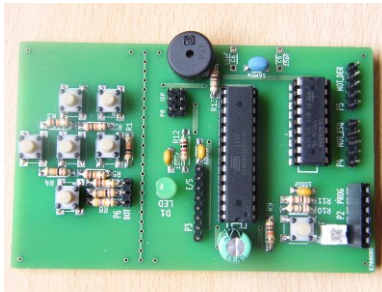
3) Para o deseño da placa usaremos como base a **Arduino Pro-Mini** (Sparkfun). Esta placa leva un conector de 6 contactos no que se enchufa un adaptador serie-usb. Este tipo de circuito non se integra na placa porque é difícil de soldar (smd) e non é imprescindible telo sempre, só cando se reprograma o micro (lembremos que estamos facendo un robot, que é un equipo autónomo e non ten que estar sempre enchufado ó ordenador).



CONTROL COMPATIBLE: CRITERIOS DE DISEÑO

4) Usaremos o contorno Arduino coas súas bibliotecas de funcións e sketches. Calquera programa feito para unha placa tipo duemilanove ou Pro-Mini servirá para ésta. Como o noso robot é unha versión do Escornabot, o mesmo sketch debería servir para os dous.

5) Faremos un circuito propio no que se integrará todo o control do robot. Non será necesario facer cableados da placa de control ó teclado, ós motores, etc. Iso evita complicacións e posibles fallos (recordemos de novo que isto é un vehículo autónomo e os cables teñen o costume de afrouxarse...).



6) Habará dous modelos de placa, un feito en placa de tiras (que haberá que cablear mediante pontes) e outro con placa de circuito impreso. A placa de circuito impreso poderá encargarse (para iso temos os arquivos gerber, noutro apartado veremos cómo e dónde) ou facerse se dispoñemos dun taller electrónico. O deseño das pistas facilitará a construción e montaxe, por tanto as pistas serán anchas, ben separadas, con taladros grandes (1mm ou máis).

7) Para a motorización usaremos o mesmo modelo que o Escornabot, o 28BYJ-48. É un motor paso a paso unipolar con reductora que se controla mediante o circuito integrado ULN2003, un para cada motor. Na nosa placa tomamos a decisión de substituílos por un único circuito ULN2803 (ou ULN2804), xa que este último ten 8 saídas de potencia e pode controlar ó mesmo tempo os dous motores. Isto reduce o tamaño da placa, elimina moitos cables e supón un pequeno aforro económico. O uso dun motor paso a paso ten vantaxes e inconvenientes, é máis difícil de controlar pero permite facer moitas aplicacións nas que se necesite un control preciso de posición ademais de movemento.



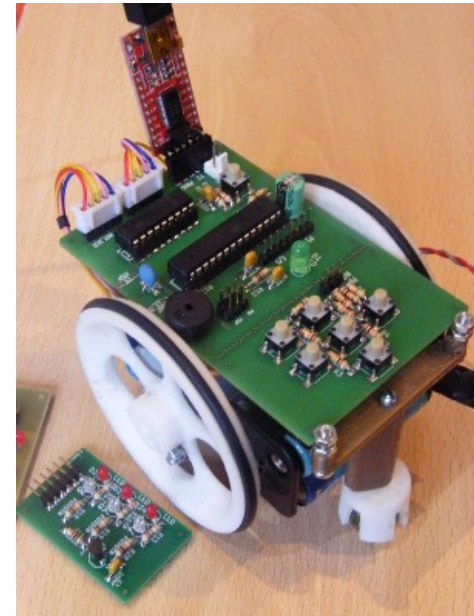
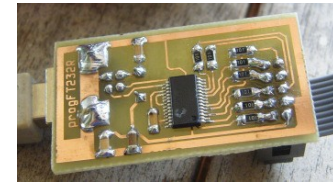
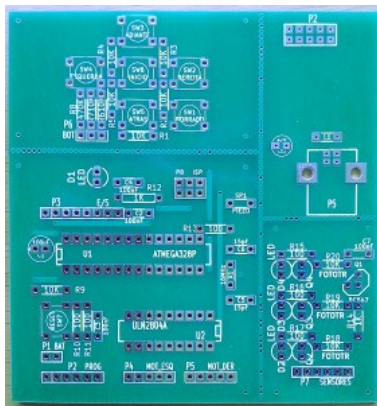
CONTROL COMPATIBLE: CRITERIOS DE DISEÑO

8) O módulo de comunicacións serie-usb pode mercarse ou facerse, temos un deseño de placa (feito para outro sistema de microcontroladores, o AT89 de Atmel) que se pode montar pero que é complicado de soldar.



9) Tamén faremos o deseño dun módulo de sensores infravermellos, que non estaba previsto non Escornabot pero é interesante para facer programas de seguemento de liñas.

10) Todas as placas se inseriron nun cadrado de 10x10cm para que saíra máis barato encargar a fabricación (moitos provedores traballan con ese tamaño). Nese cadrado está o control, o teclado, a placa de sensores e a de comunicacións serie-usb.



CONTROL COMPATIBLE: CRITERIOS DE DISEÑO

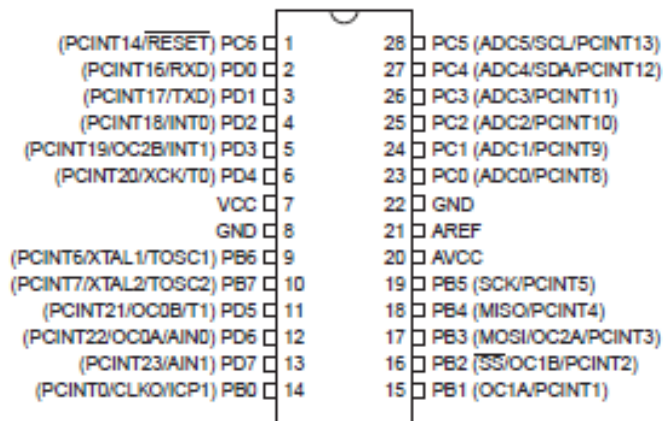
11) Por último, unha cuestión importante: a fonte de alimentación.

- Para simplificar a nosa placa non leva regulador de tensión, polo que a tensión será a mesma da pila ou do usb, dependendo se estamos en modo de programación ou autónomo.
- A tensión do usb son 5V e a das pilas pode variar entre 4 e 6V segundo o tipo de pilas, se son recargables ou non, etc. Incluso podería usarse un pack de baterías de litio (3,7-4,2V).
- Cando se conecte o adaptador usb HAI QUE DESCONECTAR A PILA, non é recomendable ter as dúas fontes ó mesmo tempo.
- Existe a posibilidade de colocar díodos nas dúas entradas de alimentación e así poderían estar conectadas ambas, pero tamén producen unha pequena caída de tensión. Esta é unha cuestión que pode revisarse nunha nova versión da placa, pero tamén se pode facer agora con pequenas modificacións (na placa de tiras é inmediato e na de circuito impreso hai que cortar as pistas nun para de puntos e soldar os díodos). Os díodos máis axeitados son de tipo Schottky (1N5819 ou similar).

TÁBOA DE CONEXIÓNS DO CONTROL ROBÓTICO

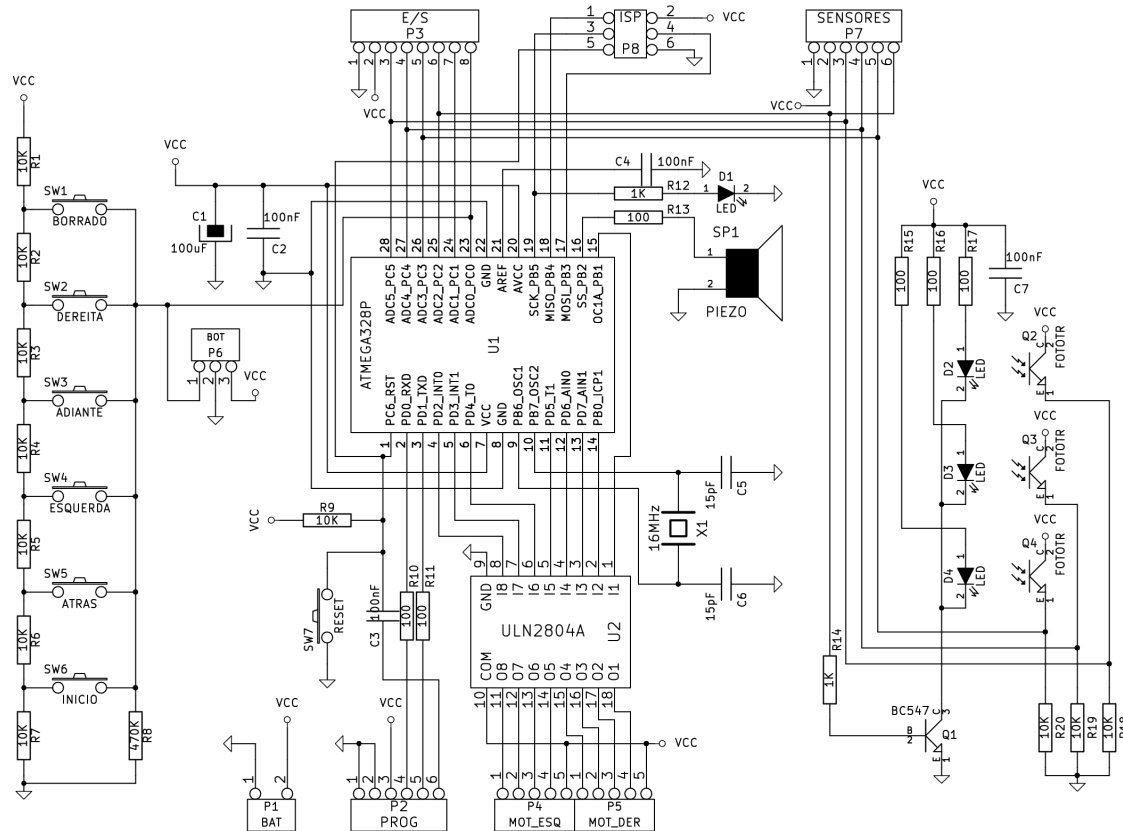
FUNCIÓN	ARDUINO	ATMEGA328P-PDIP	
	nome	nº pin	nome
motor esquerdo IN1	D2	4	PD2 (INT0)
motor esquerdo IN2	D3	5	PD3 (INT1)
motor esquerdo IN3	D4	6	PD4 (T0)
motor esquerdo IN4	D5	11	PD5 (T1)
motor dereito IN1	D6	12	PD6 (AIN0)
motor dereito IN2	D7	13	PD7 (AIN1)
motor dereito IN3	D8	14	PB0
motor dereito IN4	D9	15	PB1
altavoz (piezo)	D10	16	PB2 (SS)
led indicador	D13	19	PB5 (SCK)
botoneira analóxica	A0	23	PC0(ADC0)
led sensores	A2	25	PC2(ADC2)
fotodetector esquerda	A3	26	PC3(ADC3)
fotodetector centro	A4	27	PC4(ADC4)
fotodetector dereita	A5	28	PC5(ADC5)

CONEXIÓNS DO MICROCONTROLADOR ATMEGA328P-PU

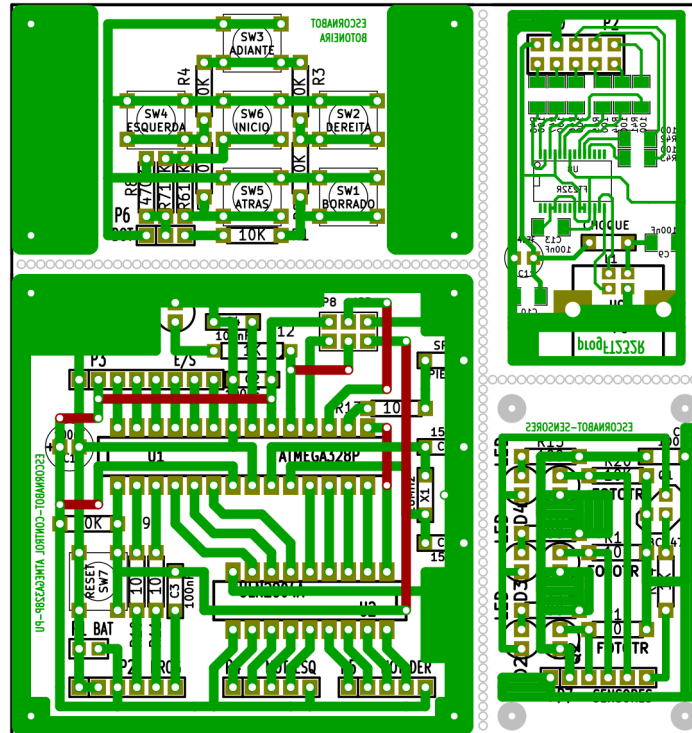


ATMEGA328P

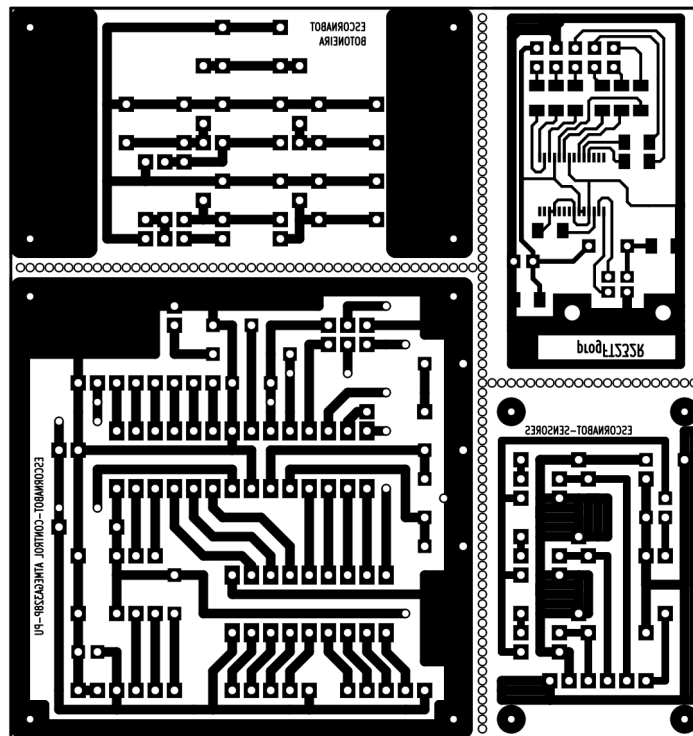
PLACA DE CIRCUITO IMPRESO: ESQUEMA ELECTRÓNICO



PLACA DE CIRCUITO IMPRESO: PLANO DE MONTAXE

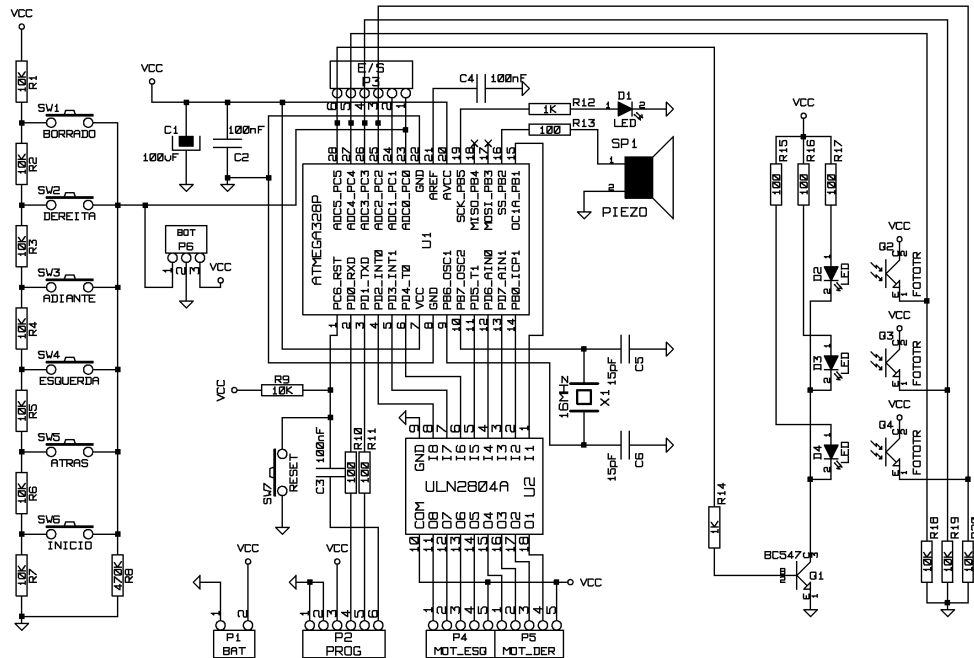


PLACA DE CIRCUITO IMPRESO: TRAZADO DE PISTAS



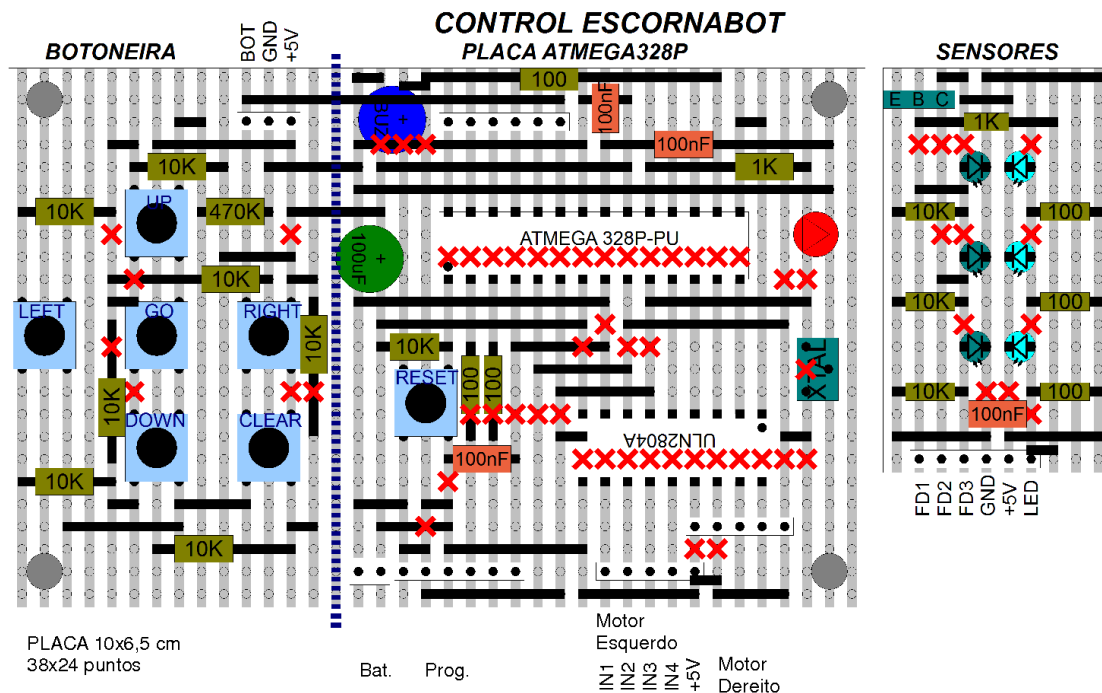
PLACA DE TIRAS: ESQUEMA ELECTRÓNICO

Este esquema é practicamente igual que o de circuíto impreso, excepto que non leva o conector de programación (bloque de 6 terminais). Tampouco hai un conector de sensores (van cableados). Ademais, no esquema está previsto que o control dos leds dos sensores se faga coa saída PC5 do micro no canto de PC2 no esquema de circuíto impreso (inda que isto pode cambiarse no cableado final).



PLACA DE TIRAS: PLANO DE PONTES E CORTES

As liñas negras indican as pontes de cable e as cruces vermellas os cortes nas tiras de cobre. Para facelas recoméndase usar unha broca de 3-4mm ou elemento similar e rotar a man facendo centro no burato. As tres placas poden separarse cortando pola liña de puntos.



GRAVACIÓN DO BOOTLOADER NUN MICRO NOVO USANDO ARDUINO COMO PROGRAMADOR

O bootloader é un programa que está sempre no microcontrolador e se comunica co contorno arduino para transferir o código ("sketch"). Os micros mercados de fábrica non o traen e hai que instalalo.

Este proceso só hai que facelo cando merquemos micros novos. Nese caso gravamos o bootloader en todos eles (leva só uns segundos) e xa quedan preparados para usar nas placas Arduino.

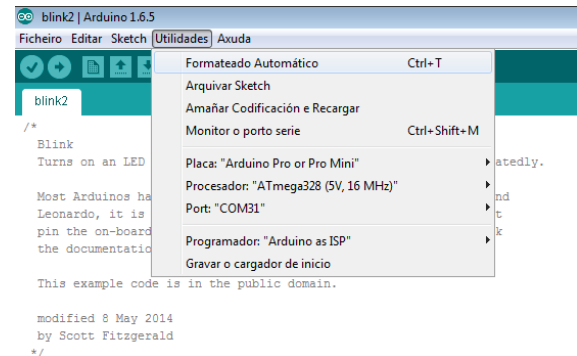
Se non dispoñemos dun gravador específico como o AVRISP pódese usar outra placa Arduino (Nano, Uno, Duemila,...) como programador ISP.

Este procedemento se explica no tutorial: <https://www.arduino.cc/en/Tutorial/ArduinoToBreadboard>

PASOS A SEGUIR

1) Configurar a placa Arduino como gravador ISP

- Seleccionar en "Utilidades" a placa da que dispoñamos.
- Abrir o sketch "Arduino as ISP", compilar e cargar na placa.



2) Configurar o tipo de Bootloader e modo de transferencia

- Seleccionar tipo de programador "Arduino as ISP".
 - Seleccionar o tipo de placa **que se usa como gravador** (por exemplo, Arduino Nano, Pro-Mini).
 - Seleccionar o micro que queremos gravar (Atmega328p, 5V, 16MHz).
- MOI IMPORTANTE: Seleccionar o micro correcto, o bootloader é distinto para cada tipo de micro!!!
- Seleccionar o porto serie no que estea instalado o adaptador (COMxx). Isto só se fai nas placas que non teñan conexión usb directa (como Pro-Mini).

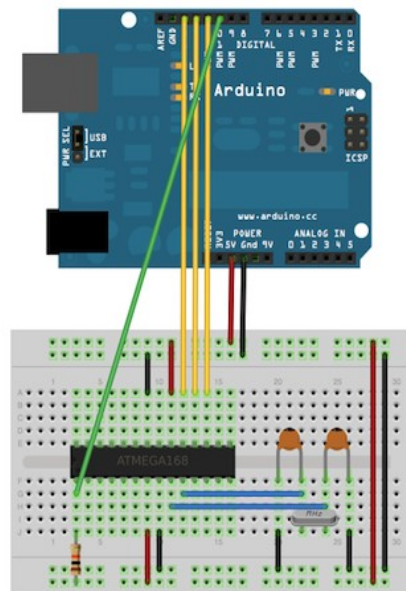
GRAVACIÓN DO BOOTLOADER NUN MICRO NOVO USANDO ARDUINO COMO PROGRAMADOR

3) Facer a seguinte montaxe nunha placa de probas ou similar. Se necesita un cristal de cuarzo con condensadores, unha resistencia e fío ríxido. Pode montarse tamén o led de control na pata 19 do micro (D13) para verificar a carga, xa que está conectado a unha das liñas de comunicacións.

Conexións:	ARDUINO	ATMEGA328P-PU
SS	D10	1 (RESET)
MOSI	D11	17 (MOSI)
MISO	D12	18 (MISO)
SCK	D13	19 (SCK)
cristal 16MHz	----	9,10
+5V	----	7, 20
GND	----	8, 22

4) Unha vez conectado todo, no menú "Utilidades" seleccionar "Gravar cargador de inicio" e esperar a que remate (son uns poucos segundos).

5) Se temos que gravar máis micros, se quita da placa o actual e se repite o proceso tantas veces como sexa necesario (non hai que volver a configurar nada).



GRAVACIÓN DUN PROGRAMA ("SKETCH") NUN MICRO CON BOOTLOADER E ADAPTADOR USB

Se a placa xa está conectada ó ordenador cun adaptador serie-usb e o micro ten o *bootloader* de Arduino o proceso é o mesmo que con calquera placa de Arduino.

Para conectar a placa e o adaptador usamos a mesma configuración que no Arduino Pro-Mini. Esta placa leva un conector de comunicacións de 6 patas que se conectan ó adaptador usb.

PROCESO DE GRAVACIÓN

Seleccionar placa Arduino Pro-Mini (sen conexión usb directa).

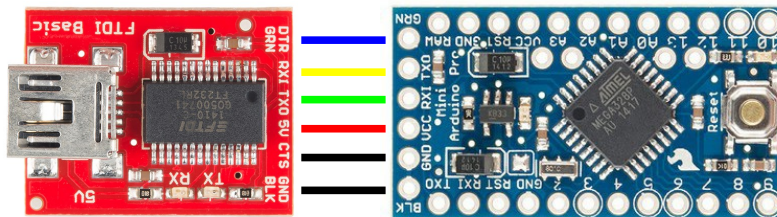
Seleccionar o tipo de micro (Atmega328P, 5V, 16MHz)

Seleccionar o porto serie no que estea instalado o adaptador (COMxx).

Unha vez axustados estes parámetros, xa pode compilarse e transferir calquera sketch usando a opción normal (Menú "Sketch", opción "Cargar").

DIAGRAMA DE CONEXIÓNS (ARDUINO PRO-MINI)

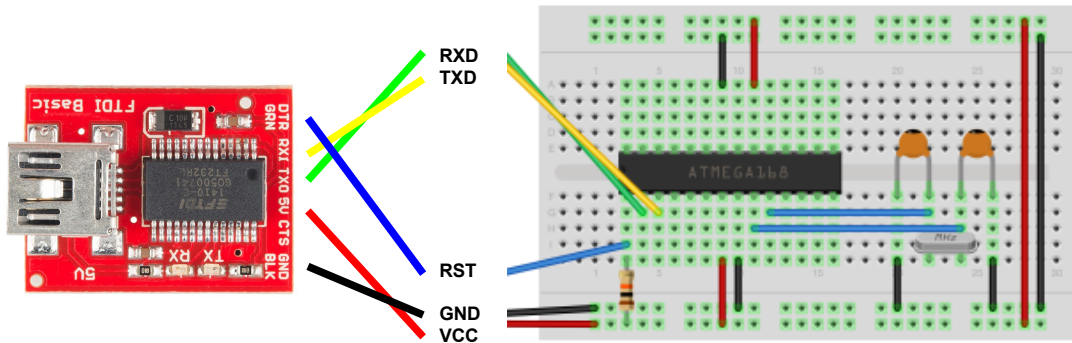
- Se queremos gravar unha placa Arduino Pro-Mini (ou compatible) a forma de conexión sería a seguinte. Son 5 liñas: GND, VCC, RXD, TXD, RST. Nos conectores hai 6 cables porque a masa está duplicada:



GRAVACIÓN DUN PROGRAMA ("SKETCH") NUN MICRO CON BOOTLOADER E ADAPTADOR USB

DIAGRAMA DE CONEXIÓNS (PLACA PROTOTIPOS)

- Se o micro está montado nunha placa de prototipos a conexión sería a seguinte. Tamén son 5 liñas (GND, VCC, RXD, TXD, RST):

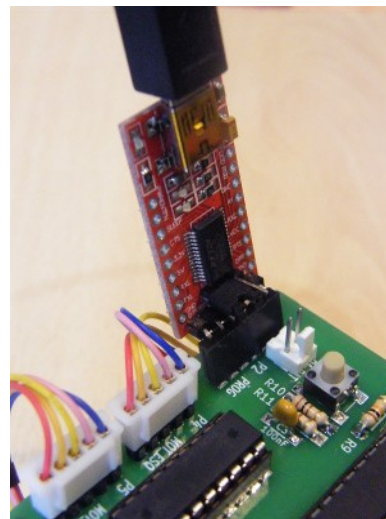


GRAVACIÓN DUN PROGRAMA ("SKETCH") NUN MICRO CON BOOTLOADER E ADAPTADOR USB**DIAGRAMA DE CONEXIÓNS (PLACA COMPATIBLE ESCORNABOT)**

- Por último, na nosa placa de control se conecta o adaptador directamente ou cun cable, dependendo do tipo de conector que puxeramos na placa (macho ou femia). O esquema de conexións é o seguinte (incluíndo tamén a opción de adaptador con conector IDC-10).

CONEXIÓNS

Arduino PRO-MINI (ou compatible)		FTDI-IDC10		ATMEGA328P-PU	
pin	nome	pin	nome	pin	nome
1	GND	4	GND	8,22	GND
2	GND	----	----	----	----
3	VCC	2	VCC	7,20	VCC
4	RXI	3	TXD	2	RXI
5	TXO	10	RXD	3	TXO
6	DTR	1	DTR	1	RESET



NOTA: As liñas RXI e TXO se consideran desde o microcontrolador, por ese motivo RXI se conecta a TX na placa FTDI (o adaptador usb envía datos e chegan á liña de recepción do microcontrolador).

OUTRAS OPCIÓNS DE GRAVACIÓN

OUTROS ADAPTADORES QUE SE PODEN USAR

- Pode usarse un cable de comunicacións serie-usb que teña saídas TTL, sempre que estean as conexións DTR, RX, TX, VCC, GND. Hai moitos no mercado.
- Pode usarse unha placa Arduino normal como adaptador serie-usb, nese caso hai que quitar o micro. Non valen todas, só as que teñan zócalo e se poida quitar o circuito integrado.
- Arduino vende unha placa especial chamada ArduinoISP (non confundir co sketch "ArduinoasISP"). Esta placa fai a mesma función que o adaptador usb.

GRAVACIÓN DE SKETCHES SEN BOOTLOADER USANDO ARDUINO COMO GRAVADOR ISP

- Tamén se pode usar a placa Arduino como gravador para cargar directamente sketches sen usar o bootloader.
- Neste caso en vez de "cargar" hai que escoller "cargar usando o programador".
- Hai que acordarse de seleccionar o tipo de placa destino (non ten que ser a mesma do programador) antes de compilar o sketch. Tamén hai que especificar correctamente o tipo de micro e velocidade.
- A forma de conexión é igual que para gravar o bootloader.
- Esta opción é interesante se queremos usar todo o espazo de memoria dos micros, xa que pode eliminarse o bootloader.

SENSORES OPTOELECTRÓNICOS

Para completar o robot tamén se fixo o deseño dun circuito de sensores reflectivos con diodos led e fototransistores. Isto permite facer programas de seguemento de liñas marcadas no chan ou detección de obstáculos.

No mercado hai moitos sensores deste tipo para robótica. Como exemplo, está o TCRT5000, que se usa en robots como os Bitbloq (<http://www.elec freaks.com/estore/octopus-hunt-sensor.html>).

- Prezo 0,6 euro aprox. Máis barato en versión TCRT5000L (patas longas)
- Distancia detección 2-15mm (20%)
- Intensidade led < 60mA (tip. 20mA), tensión led 1,25V
- Saída con fototransistor
- Códigos TCRT5000 (Farnell 1470066, RS 708-5017), TCRT5000L (Farnell 1703519, RS 818-7524)



Outros sensores son o CNY70 (distancia 5mm, funciona ata 1cm, código Farnell 1470063) ou o QRD1114 (distancia 1,5mm, código Farnell 1467858).

Moitos destes sensores teñen saída dixital polo que só poden diferenciar se a luz detectada é superior ou inferior a un valor prefixado (umbral).

Como o microcontrolador Atmega ten 6 canles de entrada analóxicas é posible utilizalos para facer unha lectura máis precisa da luz recibida, con valores numéricos entre 0 (negro) e un máximo (por exemplo 1024=branco). Con esta información o algoritmo de desprazamento do robot pode mellorarse e suavizar os movementos.

Tamén optamos por usar un diodo led e fototransistor separados (en lugar dun módulo como o TCRT5000), isto dá máis posibilidades á hora de escoller os compoñentes e tamén se pode modificar a súa posición na placa, orientación (por exemplo para incrementar a distancia de detección).

SENSORES OPTOELECTRÓNICOS: VISIBLES OU INFRAVERMELLOS?

- Hai materiais como moitos plásticos que son transparentes ou reflectantes ó infravermello e teñen o comportamento contrario en visible. Por exemplo os plásticos transparentes de invernadoiro reflicten o infravermello, e as radiografías que parecen obscuras son casi transparentes ó infravermello (por iso non se deben usar para ver o sol!!!). Por tanto é doado equivocarse e pensar que un material é "negro" en visible cando para infravermello non o é. Habería que usar unha cámara infravermella para asegurarse, pero iso non é fácil e menos nunha carreira de robots...
- O motivo de usar sensores infravermellos é que non lles afecte a luz ambiente, especialmente en interior con iluminación fluorescente.
- Iso pódese acadar tamén facendo unha calibración previa: hai que facer dúas lecturas consecutivas, cos leds apagados e acendidos, e restar os valores. Así se elimina o efecto ambiental e se podería usar luz visible (sempre que non varíe moito).
- Outro motivo para usar visible é que facilita o axuste de iluminación e orientación dos leds (por exemplo, para comprobar que iluminan diante dos sensores e non están desviados).
- O diodo led neste caso pode ser calquera (mellor vermello ou laranxa para aproveitar a sensibilidade e porque as luces interiores tenden a ser azul-verdes).

Para ter as dúas opcións:

- Fototransistor SFH310-FA-2/3 (infravermello, 3mm, +/-25°, fotocorrente máx. 2-3mA, RS 654-8744).
- Led: Kingbright L-34F3BT (infravermello, 3mm, +/-25°, 1.5V, RS 860-9560).
- Fototransistor visible: SFH-309-5/6 (3mm, +/-12°, RS 665-5381); SFH 310-2/3 (3mm, RS 654-8542), SFH 300-3/4 (5mm). NOTA: os modelos SFH... FA levan filtro infravermello, os que non levan FA son visibles.
- Led visible alta luminosidade: Vishay TLHA44R1S2 (625nm, +/-30°, 2,6V, RS 773-4038)

Todos estes fotodiodos e fototransistores custan 0,15-0,20 euros cada un.

CIRCUITOS IMPRESOS: CÓMO E DÓNDE ENCARGALOS?

Para facer o deseño dun circuíto impreso hai moitos programas, como o Kicad (libre) ou o Eagle (licencia gratuita limitada).

A partir destes programas pode facerse a placa no taller ou encargala a un proveedor externo. Para iso se necesita un xogo de arquivos de datos que conteñen toda a información das diferentes capas do circuíto (pistas, serigrafía, taladrado, máscaras de soldadura). O formato máis utilizado chámase Gerber.

ARQUIVOS GERBER EN KICAD

Tutorial: <http://blog.iteadstudio.com/how-to-generate-gerber-files-from-kicad/>

ou: <https://code.google.com/p/opensourcious/wiki/KiCADTutorialCreatingGerberFiles>

NOTA: KICAD 2013 crea correctamente os arquivos gerber para os provedores, os anteriores non.

na opción Plot (Trazar) se exportan os arquivos:

capa	extensión
bottom layer (pistas)	.GBL
top layer (comp)	.GTL
bottom silk	.GBO
top silk	.GTO
bottom solder mask	.GBS
top solder mask	.GTS
edges	.GBR

ademáis hai que crear o arquivo de taladrado .DRL (e logo cambiarlle o nome a .TXT para moitos provedores). Hai que seleccionar as opcións unidades: pulgadas, formato: suppress leading zeros, precisión: 2/3, orixe: absoluto, desmarcar mirror y axis. Hai que axustar diámetro de taladros a 0,040" (1mm) ou similar, tamén se pode facer cando se crea o arquivo de taladrado.

CIRCUITOS IMPRESOS: CÓMO E DÓNDE ENCARGALOS?

Hai fabricantes de circuitos impresos europeos, norteamericanos e en China, India,...

En xeral hai que ter os arquivos gerber (empaquetados nun .zip), cargalos na páxina web e en moitos casos pódense verificar antes de facer o pedido (tamén calcular o prezo). Hai que poñer as extensións correctas nos arquivos, en moitos casos o arquivo de taladrado ten que ser .TXT e o de bordes (edges) .GKO. O resto quedan igual (GBL, GTL, GBO, GTO, GBS, GTS).

En Europa:

EUROCIRCUITS: <http://www.eurocircuits.com/> Caro, pero emite factura con IVE.

OLIMEX: <https://www.olimex.com/> Máis económico.

EEUU: OSH PARK <http://oshpark.com> Para múltiplos de 3 placas ou tamaño total de 150" (múltiplos de 10).

CHINA: Calidade aceptable, son moito máis baratos, tardan bastante en entregar as placas (1-2 meses).

SEEDSTUDIO <http://www.seedstudio.com/service/index.php?r=pcb> Económico, ten unha utilidade de verificación que permite cargar e visualizar os gerber antes de facer o pedido.

ITEADSTUDIO <http://www.itead.cc/open-pcb/pcb-prototyping.html> Económico, non ten utilidade de verificación previa, só se pode ver despois de facer o pedido.

OUTROS PROVEEDORES

Adaptador serie usb FT232R:

- Amazon: <http://www.amazon.com/FT232RL-Serial-Adapter-Module-Arduino/dp/B00HSX3CXE>
- Sparkfun: <https://www.sparkfun.com/products/12731>
- outro de sparkfun: <https://www.sparkfun.com/products/9718>
- Lightinthebox: http://www.lightinthebox.com/es/ft232rl-usb-a-serie-232-modulo-adaptador-ttl-para-funduino-blue-3-3-5v_p1141439.html
- Aliexpress: <http://es.aliexpress.com/item/FT232RL-FTDI-USB-to-TTL-Serial-Adapter-Module-for-Arduino-Mini-Port-3-3V-5V/2043815349.html>
- Ebay: http://www.ebay.es/sch/i.html?_from=R40&_sacat=0&_nkw=ft232rl+usb+to+serial+adapter&_sop=15

Cables USB-A mini usb-B: <http://es.aliexpress.com/item/2015-Seal-Mini-USB-5-Pin-sync-cable-USB-DATA-and-charger-cable-v3-USB-2/32570474768.html>

A close-up photograph of a white, three-fingered robotic hand. The hand is holding a small, bright yellow, glowing object. The entire scene is bathed in a strong red light, which creates a dramatic, high-contrast effect. The background is dark and out of focus.

FIN DA PRESENTACIÓN