Universidad de Granada

decsai.ugr.es

Teoría de la Información y la Codificación

Grado en Ingeniería Informática

Hardware de lectura de códigos 1-D y 2-D

DECSAI

Departamento de Ciencias de la Computación e Inteligencia Artificial

UNIVERSIDAD DE GRANADA

Teoría de la Información y la Codificación

Grado en Ingeniería Informática

1. Códigos de barras
2. El sensor DE2120. Especificaciones
3. Configuración inicial
4. Conexión a PC
5. Biblioteca Python para DE2120
6. Trabajo a realizar

DECSAI

- Un código de barras sirve para representar información numérica.
- Se compone por una secuencia de barras negras y blancas, de distinto grosor. Por tanto, es un código unidimensional (1-D).
- El grosor y cantidad de cada barra juega un papel fundamental, y de ello depende la información codificada.
- Permite detectar errores de lectura del código (no corregirlos) mediante el uso de un dígito de control, de modo que se pueda saber con alta probabilidad cuándo el código percibido por el sensor se ha leído correctamente.

- Existen múltiples estándares:
  - EAN-13 (European Article Number)
  - UPC-A
  - UPC-E
  - etc-

- Hoy día, existe una única organización llamada **GS1**, encargada de organizar los protocolos de codificación de los códigos de barras.

  Los estándares identifican tanto la composición de los mensajes del código, como su codificación gráfica, tamaño de etiquetas, etc.

- **EAN proviene de *European Article Number* Mnúmero de artículo europeo).**
- EAN es un estándar de codificación de 12 dígitos.

Se les conoce coloquialmente como ***códigos de barras*.**

En particular, para el EAN-13

- El ancho mínimo (resolución) del código de barras en el eje X es de 0.33 mm
- El alto del código debe ser de 25,9 mm

- Se compone de 12 dígitos codificados en barras de diferente grosor, más un dígito de control de errores
- Contiene barras adicionales de localización, a la izquierda (inicio), derecha (fin) y centro del código (mitad).
- Los 12 dígitos se distribuyen en 6 (mitad izquierda) y 6 (mitad derecha).

XS LLLLLL M RRRRRR E

Lectura de un código EAN-13

- El código es unidimensional. Se lee de izquierda a derecha con la siguiente distribución:

S LLLLLL M RRRRRR E

donde:

- **S** delimita el comienzo del mensaje.
- **LLLLLL** son los 6 primeros dígitos codificados.
- **M** es la delimitación del centro del código.
- **RRRRRR** son los 6 últimos dígitos codificados.
- **E** delimita el fin del mensaje.

– Codificación de un código EAN-13

- Cada dígito de los 12 posibles a codificar LLLLLLRRRRRR, se codifica con 4 barras de grosor variable: 2 negras y 2 blancas.
- Las barras (blancas o negras) tienen un ancho de 1, 2, 3 ó 4 **unidades** (donde la unidad es el grosor mínimo en milímetros establecido en el estándar).
- Se debe dejar un espacio en blanco suficientemente amplio a ambos lados del código, **antes de S y después de E**, con el objetivo de que pueda ser identificado correctamente (***zona silenciosa***).
- La zona silenciosa debe tener, a ambos lados, al menos 9 unidades de color blanco.

Codificación de un código EAN-13

- Un código EAN está compuesto de 95 unidades en total, distribuidas de la siguiente forma:
  - 84 unidades para distribuir entre los 12 dígitos
  - Cada dígito se codifica en unidades de ancho total 7. Es decir, hay 7 “huecos” que se deben distribuir entre 4 barras, dos blancas y dos negras.
  - El comienzo **S** y el final **E** utilizan 3 unidades cada uno, rellenar con el patrón negro-blanco-negro (010).
  - El centro **M** utiliza 5 unidades, marcadas como blanco-negro-blanco-negro-blanco (10101).

– Codificación de un código EAN-13

- Consideremos el código de 12 dígitos LLLLLLRRRRRR.
- El código de verificación de errores de lectura se calcula como sigue:

  - Se suman los dígitos en las **posiciones impares** (1, 3, 5, ..., 11)
  - El resultado se multiplica por 3.
  - Se suman los dígitos de las posiciones pares (2, 4, 6, 8, 10, 12).
  - Se calcula el resultado módulo 10. Sea M este resultado.
  - Si M=0, entonces X=0.
  - En otro caso, entonces X=10-M.

Codificación de un código EAN-13

- Ejemplo del cálculo del dígito de comprobación de errores de lectura. Caso de uso en un pack de 6 litros de leche Hacendado (no es EAN, pero el cálculo del código es el mismo):

Código 480000107107 . Código de verificación: 8

– Codificación de un código EAN-13

Código 480000107107 . Código de verificación: 8

- Se suman los dígitos en las posiciones impares (1, 3, 5, …, 11): 4+0+0+1+7+0= 12
- El resultado se multiplica por 3: 12*3= 36
- Se suman los dígitos de las posiciones pares (2, 4, 6, 8, 10, 12): 36+8+0+0+0+1+7= 52
- Se calcula el resultado módulo 10. M= 52%10= 2
- Si M=0, entonces X=0. ← No es el caso
- En otro caso, entonces **X=10-2=8**.

— Codificación de un código EAN-13

- Un código EAN-13 tiene paridad impar en los dígitos de la mitad izquierda, y paridad par en los dígitos de la mitad derecha. Esto **debe calcularse como**:
  - La suma del total de unidades ocupadas por barras negras en los dígitos de la izquierda debe ser un número impar.
  - La suma del total de unidades ocupadas por barras negras en los dígitos de la derecha debe ser un número par.

— Construcción de un código EAN-13

- Codificación de los elementos de localización **S, M, E**:

Construcción de un código EAN-13

- Cada dígito se codifica en 7 unidades que deben formar 4 barras blancas/negras:
- Además, la suma del número de unidades de la parte izquierda debe ser un número impar.
- La suma del número de unidades de la parte derecha debe ser un número par.

Construcción de un código EAN-13

- Para conseguir la paridad impar/par, existen 3 tipos de codificaciones diferentes para un mismo dígito (B/G/R).
  - **Codificación B:**

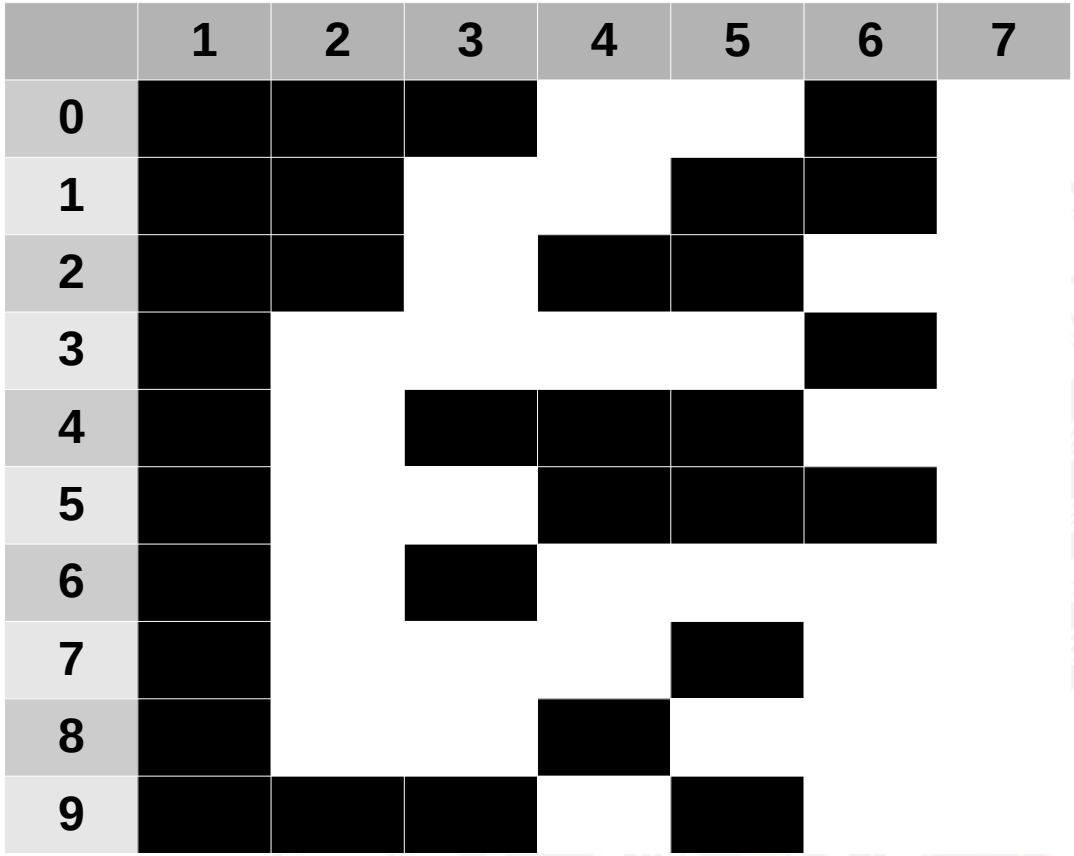
    Filas: dígitos
    Cols.: Unidades

	1	2	3	4	5	6	7
0				■	■		■
1			■	■			■
2			■			■	■
3		■	■	■	■		■
4		■				■	■
5		■	■				■
6		■		■	■	■	■
7		■	■	■		■	■
8		■	■		■	■	■
9				■		■	■

Construcción de un código EAN-13

- Para conseguir la paridad impar/par, existen 3 tipos de codificaciones diferentes para un mismo dígito (B/G/R).
  - **Codificación R:**

    Filas: dígitos
    Cols.: Unidades



Construcción de un código EAN-13

- Para conseguir la paridad impar/par, existen 3 tipos de codificaciones diferentes para un mismo dígito (B/G/R).
  - **Codificación G:**

    Filas: dígitos
    Cols.: Unidades

	1	2	3	4	5	6	7
0		■			■	■	■
1		■	■			■	■
2			■	■		■	■
3		■					■
4			■	■	■		■
5		■	■	■			■
6					■		■
7			■				■
8				■			■
9			■		■	■	■

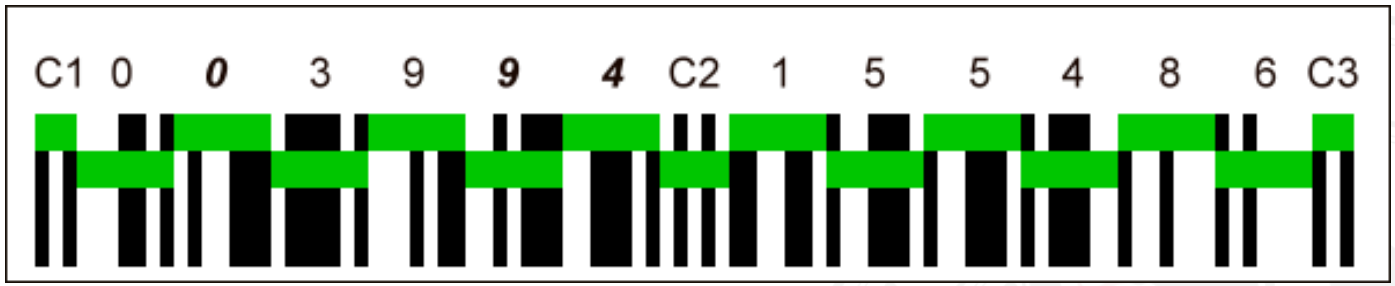
Construcción de un código EAN-13

- La elección de una codificación de dígito u otra se asigna según el valor de control de errores (paridad), de la siguiente forma:

Valor paridad	LLLLLL	RRRRRR
0	BBBBBB	RRRRRR
1	BBGBGG	RRRRRR
2	BBGGBG	RRRRRR
3	BBGGGB	RRRRRR
4	BGBBGG	RRRRRR
5	BGGBBG	RRRRRR
6	BGGGBB	RRRRRR
7	BGBGBG	RRRRRR
8	BGBGGB	RRRRRR
9	BGGBGB	RRRRRR

– Construcción de un código EAN-13

- Un ejemplo de la codificación:
  - Valor de control de errores (paridad): 4 → BGBBGG RRRRRR



Fuente:
https://en.wikipedia.org/wiki/International_Article_Number

¿Porqué tanta prevención contra errores?

- Los mecanismos de paridad par-impar detectan un número impar de errores en lecturas de 1 unidad del código.
- Además, permiten a los lectores de códigs poder identificar si el código está en posición correcta, o al revés.

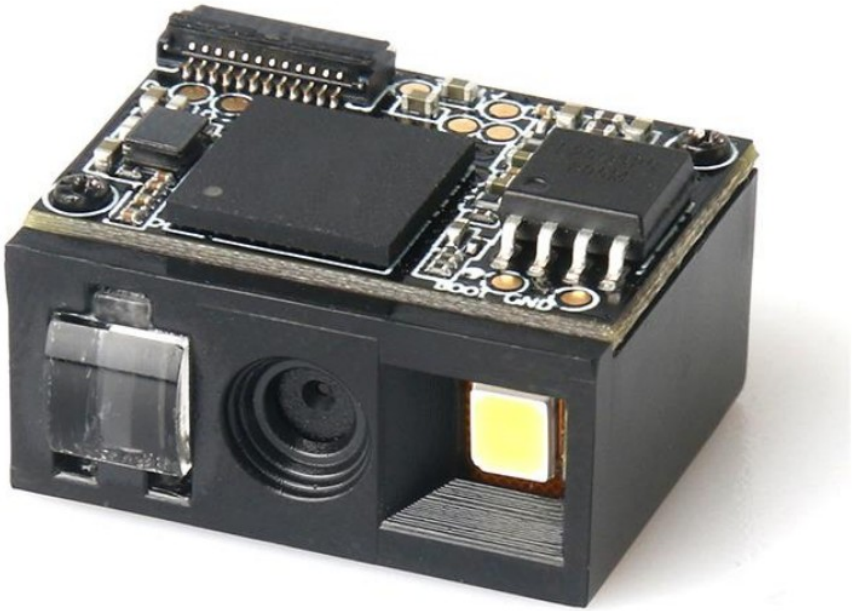
  La codificación de S, M y E permite al lector reconocer dónde comienza y acaba el código, y ajustarse para poder reconocerlo ante lecturas en ángulos no perpendiculares al código (por ejemplo, que el lector se sostenga en la mano de forma oblicua).

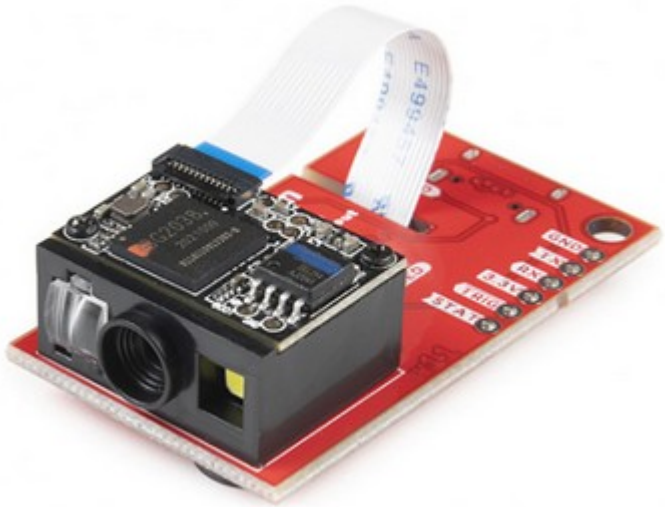
- El dígito de control permite verificar que el código se ha leído correctamente cuando es procesado directamente por un humano, y también seleccionar un método de codificación único para escribir el código de barras.

1. Códigos de barras
2. El sensor DE2120. Especificaciones
3. Configuración inicial
4. Conexión a PC
5. Biblioteca Python para DE2120
6. Trabajo a realizar

– Sensor DE2120

- Escáner de códigos de barras empotrado con tecnología de reconocimiento de imágenes CMOS.
- Equipado con un sistema de recuperación de información de imágenes. Esto permite usarlo tanto para leer códigos 1-D (códigos de barras) como 2-D (QR, por ejemplo).
- Tiene soporte para interfaz mediante USB, TTL, y otras





– Aplicaciones

- Lockers
- Investigación médica
- Máquinas de lotería
- Desarrollo de microcontroladores
- Lectura de cupones/tickets
- Pago a través de escaneo de códigos QR
- Máquinas de vending...

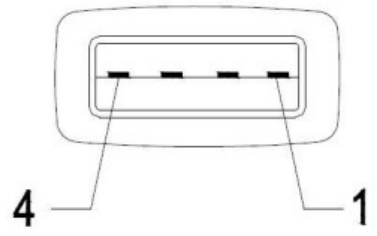
– Conexiones

PIN#	Signal	Type	Definition
1	-	-	N/A
2	VCC	P	Power supply 3.3
3	GND	P	Ground
4	RXD	Input	TTL-RS232 receive
5	TXD	Output	TTL-RS232 send
6	D-	Input/Output	USB D- signal
7	D+	Input/Output	USB D+ signal
8	-	-	N/A
9	BUZ	Output	Buzzer Output, free Low level
10	LED	Output	Decoding LED input signal, low level instruction reading codes
11	-	-	N/A
12	TRIG	Input	Trigger pin, keep low power level can trigger decoding.

– Conexión por USB:

USB

Pin No.	Function
1	Vcc
2	D-
3	D+
4	GND



– Rendimiento:

Scan Type	CMOS
Light Source	Red Light LED 625±10nm（aim), 5600K LEDs（ Lighting)
CPU	32-bit
Resolution	640*480
Reading precision	≥4mil/0.1mm(PCS90%,Code 39)
Decoding speed	25CM/S
Depth of field	25mm-400mm
Scan Mode	Manual ,Continuous, Auto Sense, command control
Scan angle	*Test Conditions： CODE39,10mil/0.25mm,PCS90%*
	Pitch:±45°
	Roll: ±360°
	Skew: ±40°
Print Contrast Signal	≥25%
Ambient Light	Dark environment, indoor natural light

El sensor DE2120 tiene diversos modos de funcionamiento

- Iluminación propia ON/OFF
- Avisos sonoros ON/OFF
- Selección de interfaz de comunicación (USB, RS232...)
- Etc.

Configuración simple mediante escaneo de códigos de barras

- Indicaciones en el manual de configuración.

– Restaurar configuración a valores de fábrica

- Basta con escanear el siguiente código:

Restore default settings

– Volumen de indicadores de sonido

- Cuatro niveles: Pasivo bajo/medio/alto, y activo medio. Escanear:

Active Medium Frequency

Passive low frequency

* Passive Medium Frequency

Passive High Frequency

– Sonido tras correcta decodificación:

Sonido ON/OFF

*Beep after Good Decode

Do Not Beep after Good Decode

– Sonido al inicio del dispositivo

Sonido ON/OFF

*Enable Startup Beep

Disable Startup Beep

– Selección de idioma:

Nos centraremos en Español

– Selección de comunicación por USB modo serie:

– Modos de lectura

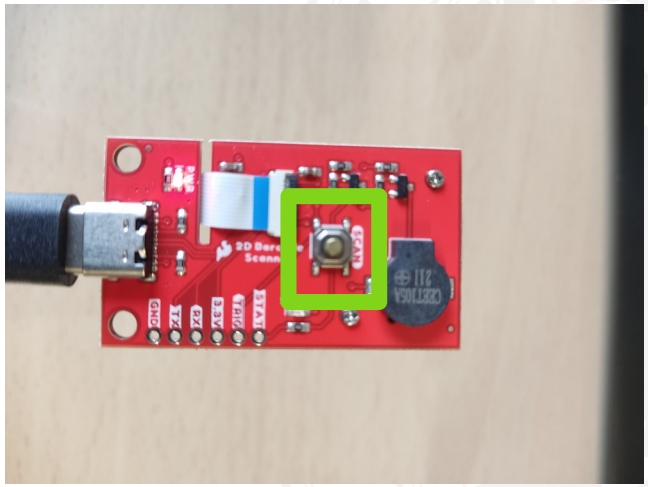
Trigger mode (sólo lee cuando se presiona el botón del dispositivo)

1. Códigos de barras
2. El sensor DE2120. Especificaciones
3. Configuración inicial
4. Conexión a PC
5. Biblioteca Python para DE2120
6. Trabajo a realizar

- En primer lugar, se debe conectar el cable USB-C al dispositivo.
- Seguidamente, el extremo USB se conecta al PC.

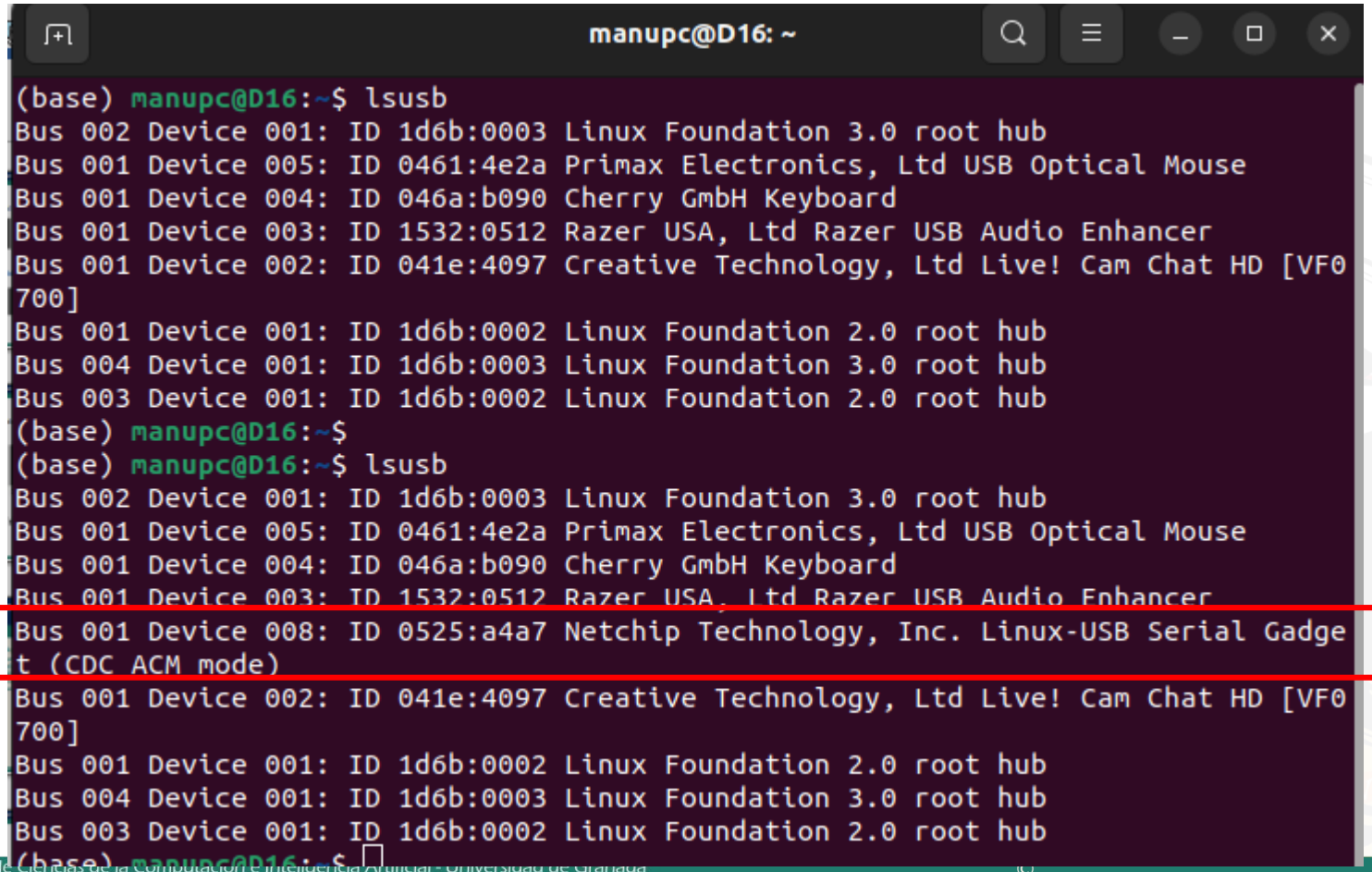
- **El siguiente paso es establecer la configuración inicial, leyendo los códigos de barras anteriores mostrados en estas diapositivas.**
- **Se debe, como mínimo, establecer la comunicación por USB serie (USB-COM)**

  - Para ello: Presionar el botón del dispositivo mientras se apunta la cámara para leer el código de barras deseado.



– Conexión en LINUX

– Si todo se hizo correctamente, se debería poder listar el dispositivo mediante el comando ***lsusb*** en consola.



– Conexión en LINUX

– Podemos saber en qué puerto COM está, listando la carpeta ***/dev/ttyACM****:

En el caso del ejemplo: ttyACM0

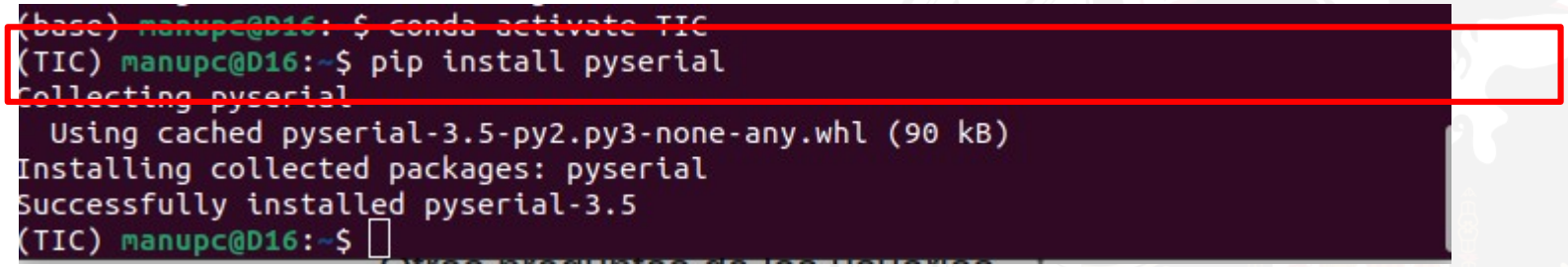
Usaremos este puerto para conectarnos al dispositivo.

1. Códigos de barras
2. El sensor DE2120. Especificaciones
3. Configuración inicial
4. Conexión a PC
5. Biblioteca Python para DE2120
6. Trabajo a realizar

- La biblioteca se proporciona con los archivos de apoyo a las prácticas, fichero *DE2120_Py-main.zip*.
- Se requiere también la instalación de la biblioteca PySerial:

  pip install pyserial

  - Se recomienda jupyter notebook para realizar las prácticas.



- **Web de la biblioteca:**
  - https://github.com/sparkfun/DE2120_Py

- **Instalación:**
  - pip install de2120-barcode-scanner , o también:
  - python setup.py install

DE2120_Py

qwiic python™

python 2.7 | 3.5 | 3.6 | 3.7 issues 0 open docs passing license MIT Follow @sparkfun 154k

Python module for the SparkFun 2D Barcode Scanner Breakout - DE2120.

This python package is a port of the existing SparkFun DE2120 Arduino Library

Contents

- Supported Platforms
- Dependencies
- Installation
- Documentation
- Example Use

- **API: situada en la plataforma ReadTheDocs:**
  - https://de2120-py.readthedocs.io/en/latest/apiref.html#de2120-barcode-scanner

de2120_barcode_scanner

Python module for the 2D Barcode Scanner.

This python package is a port of the exisiting [SparkFun DE2120 Arduino Library] (https://github.com/sparkfun/SparkFun_DE2120_Arduino_Library)

class de2120_barcode_scanner.DE2120BarcodeScanner(*hard_port=None*) [source]

Initialize the library with the given port.

Parameters: **hard_port** – The port to use to communicate with the module, this is a serial port at 9600 baud rate.

Returns: The DE2120BarcodeScanner object.

Return type: Object

USB_mode(*mode*) [source]

Enable USB communication and set the mode. THIS WILL MAKE THE MODULE UNRESPONSIVE ON COM PORT

Parameters: **mode** – string defining what USB mode to set the module in. Valid arguments are “KBD”, “HID”, “232”.

Returns: true if the command is successfully sent, false otherwise

Return type: bool

- **Estudiar la documentación de la biblioteca**
- **Realizar un programa en Python capaz de leer códigos 1-D en cualquier formato, usando el sensor DE2120.**
- **El programa debe ser capaz de devolver por pantalla el código leído.**
- **Ampliar el programa anterior para leer cualquier código 1-D y 2-D (por ejemplo, QR).**
- **Se puede hacer uso de cualquier generador de códigos QR y de barras de Internet para la realización de las pruebas.**

- **Entregas:**
  - **Los programas requeridos.**
  - **Una pequeña memoria de prácticas describiendo las componentes de la biblioteca y una descripción de las funciones utilizadas en el programa.**
  - **Ejemplos de uso y pruebas realizadas.**

– Evaluación:

- **Se evaluará la competencia y capacidad de uso del hardware DE2120, y la creación de programas que hagan interfaz con él.**
  - Descripción de las funciones de la API usadas: 20%
  - Programa para leer códigos 1-D: 60%
  - Programa para leer códigos 2-D: 20%

  - La calificación final será numérica en el rango 0-10, calculada como la ponderación de los ítems previamente indicados.

Universidad de Granada

decsai.ugr.es

Teoría de la Información y la Codificación

Grado en Ingeniería Informática

Hardware de lectura de códigos 1-D y 2-D

DECSAI

Departamento de Ciencias de la Computación e Inteligencia Artificial