Semana 6.
Sensores Digitales I2C para Sistemas Embebidos

SISTEMAS EMBEBIDOS

(EMBEDED SYSTEMS)

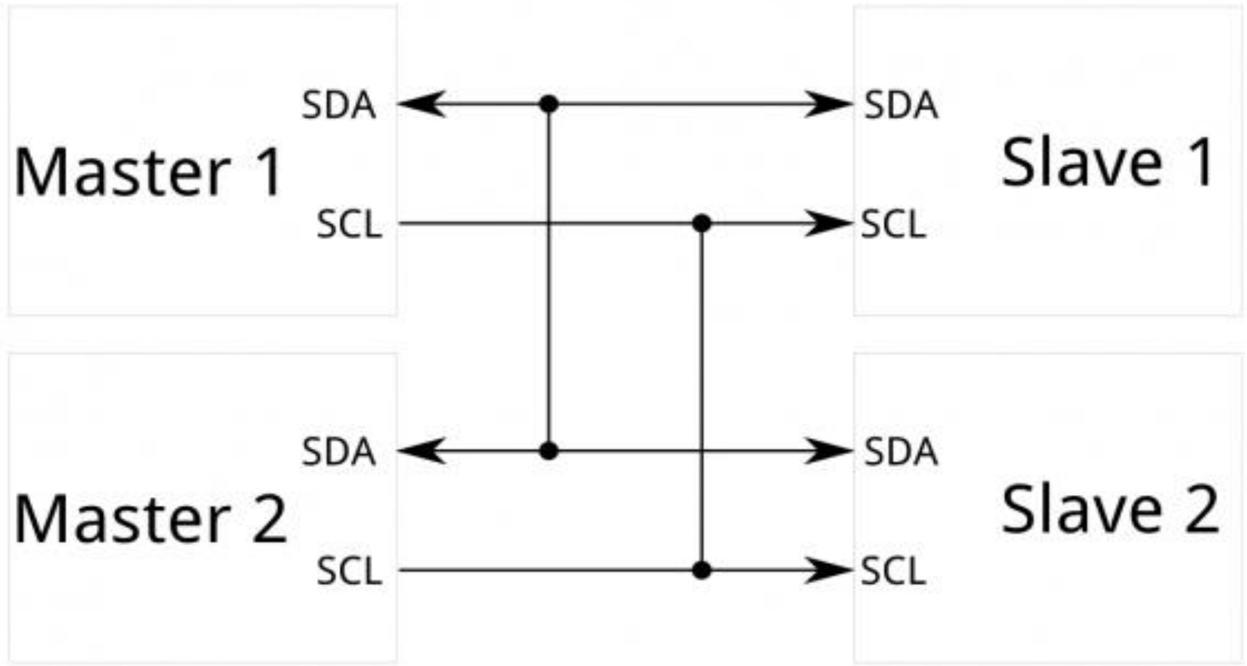
Grado Dual en Industria Digital

Campus Vitoria

Curso 2020-2021

Deusto
Facultad de Ingeniería
Ingeniaritza Fakultatea

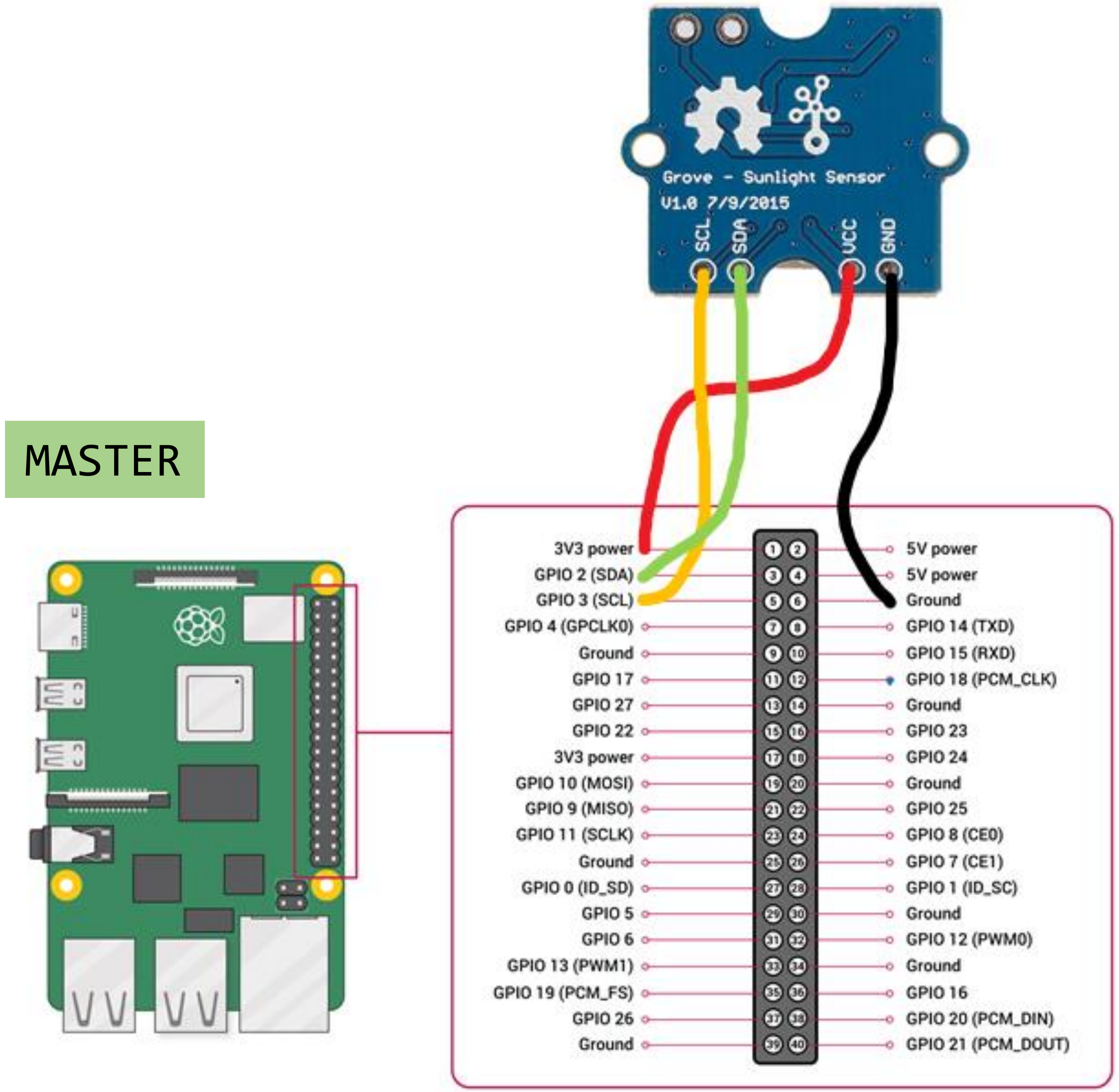
Protocolo I2C



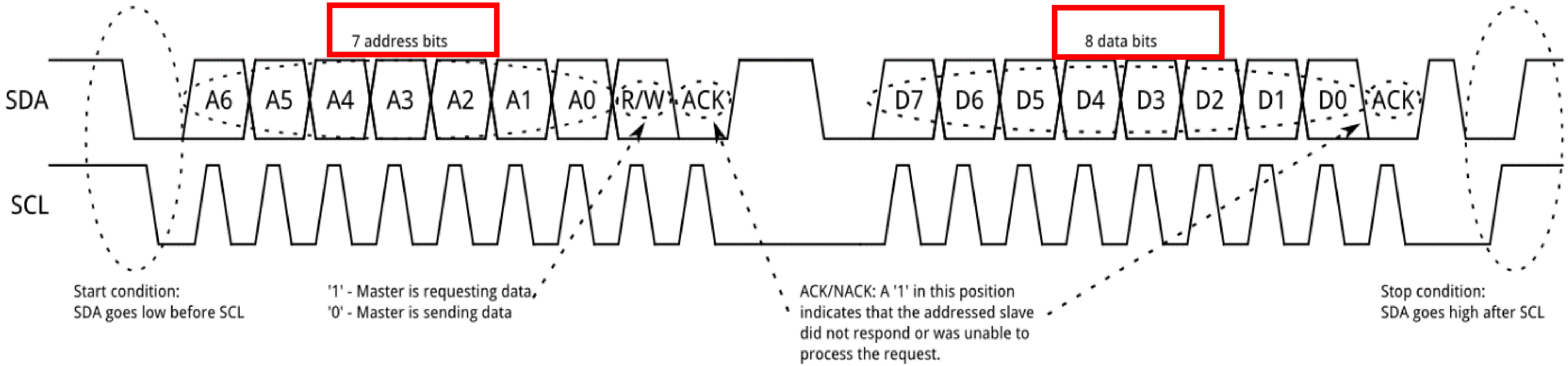
I2C conexiones

SLAVE

MASTER



I2C Transactions



I2C con python

Paso 1: Habilitar I2C en la RPi

a) desde la terminal con `$ sudo raspi-config`

b) desde la interfaz gráfica (menú inicio-configuración)

Paso 2: En el programa en Python, importar la librería smbus

Nota: si no está instalada, usad `$pip3 install smbus2`

```
import smbus
```

Documentación oficial: https://pypi.org/project/smbus2/

Deusto
Facultad de Ingeniería
Faculty of Engineering

I2C con python: Uso de smbus

Documentación oficial: https://pypi.org/project/smbus2/

Lectura y/o escritura: se basa en lectura y escritura de bytes en **registros** específicos. Los registros se referencian mediante su dirección en hexadecimal

I2C con python: Uso de smbus

Documentación oficial: https://pypi.org/project/smbus2/

Ejemplo de lectura en bloques de 1 byte

```
# Leer 3 bytes, del dispositivo con dirección I2C 0x29, del registro con dirección 0x80

data = bus.read_i2c_block_data(0x29, 0x80, 3)
```

Deusto
Facultad de Ingeniería
Faculty of Engineering

I2C con python: Uso de smbus

Documentación oficial: https://pypi.org/project/smbus2/

Ejemplo de escritura

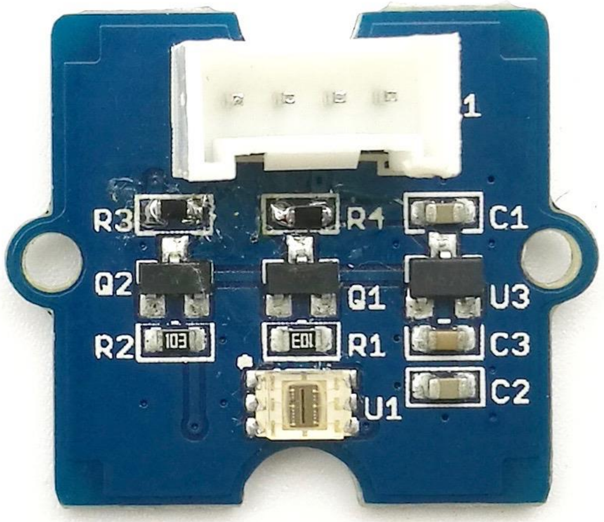
```
# Escribir un byte (data) en el dispositivo con dirección I2C 0x29, en el registro con dirección 0x80

bus.write_byte_data(0x29, 0x80, data)
```

Digital Light Sensor -- TSL2561

- Transforma la intensidad luminosa en una señal digital

- Contiene 2 diodos sensibles a diferentes longitudes de onda permite detecar luz en el espectro complento, y en el infrarojo.



•Selectable detection modes
•High resolution 16-Bit digital output at 400 kHz I2C Fast-Mode
•Wide dynamic range: 0.1 - 40,000 **LUX**

¿Cómo se mide la Luminosidad?

Lux

Lighting condition	From (lux)	To (lux)	Mean value (lux)	Lighting step
Pitch Black	0	10	5	1
Very Dark	11	50	30	2
Dark Indoors	51	200	125	3
Dim Indoors	201	400	300	4
Normal Indoors	401	1000	700	5
Bright Indoors	1001	5000	3000	6
Dim Outdoors	5001	10,000	7500	7
Cloudy Outdoors	10,001	30,000	20,000	8
Direct Sunlight	30,001	100,000	65,000	9

¿Cómo se mide la Luminosidad?

Illuminance (Iluminancia o nivel de iluminación: cantidad de luz relativa al tamaño de la superficie irradiada.
Se mide en LUX
1 LUX = 1 lumens/s^2

Lumens: cantidad de luz (flujo luminoso)

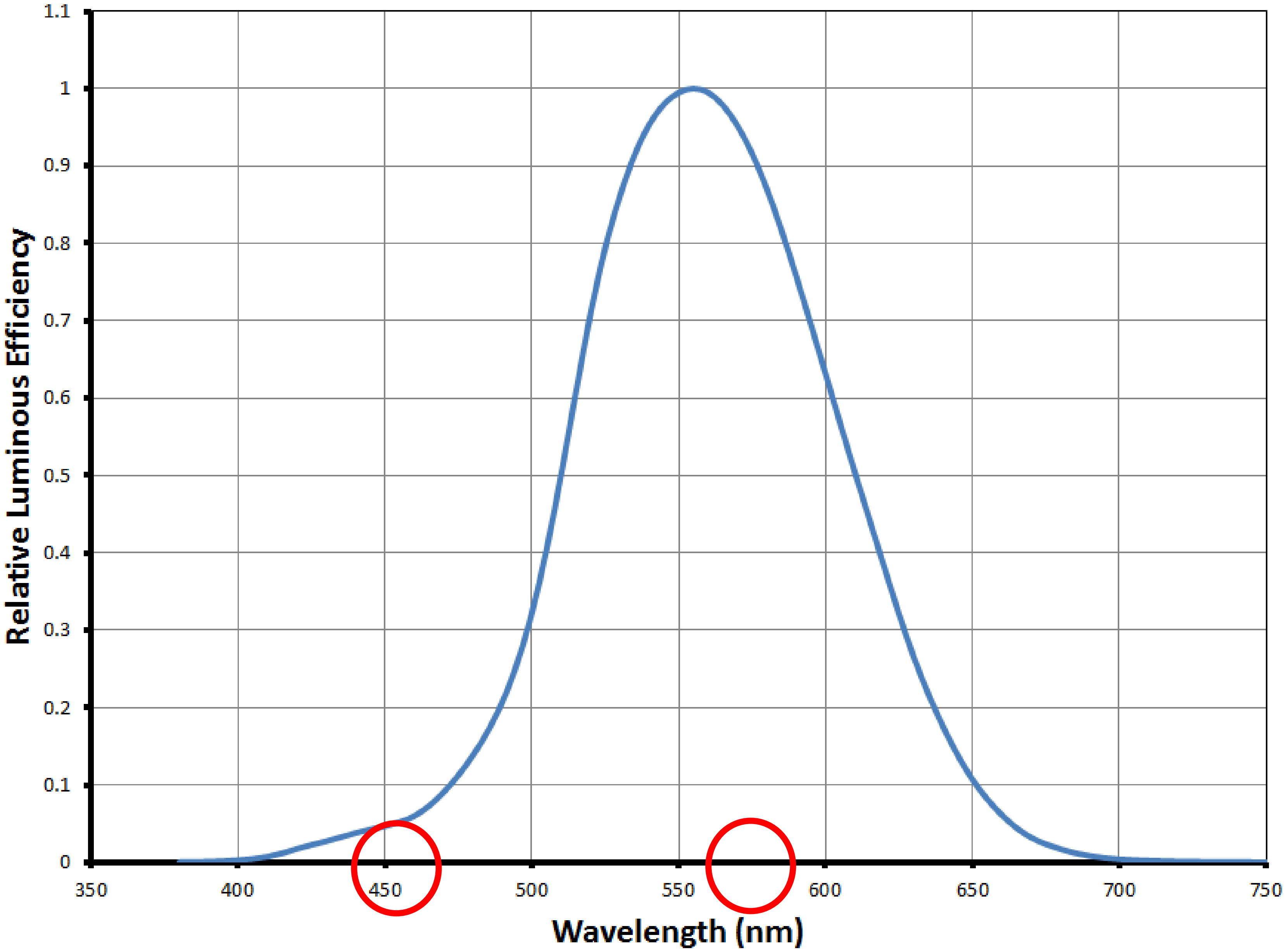
LUX: cantidad de luz por metro cuadrado

Luminosidad y longitude de onda

Si tuvieras 2 fuentes de luz, una a 555nm y otra a 450 nm, emitiendo la misma radiación.

¿Cuál de ellas interpretaría un humano cómo más intensa?

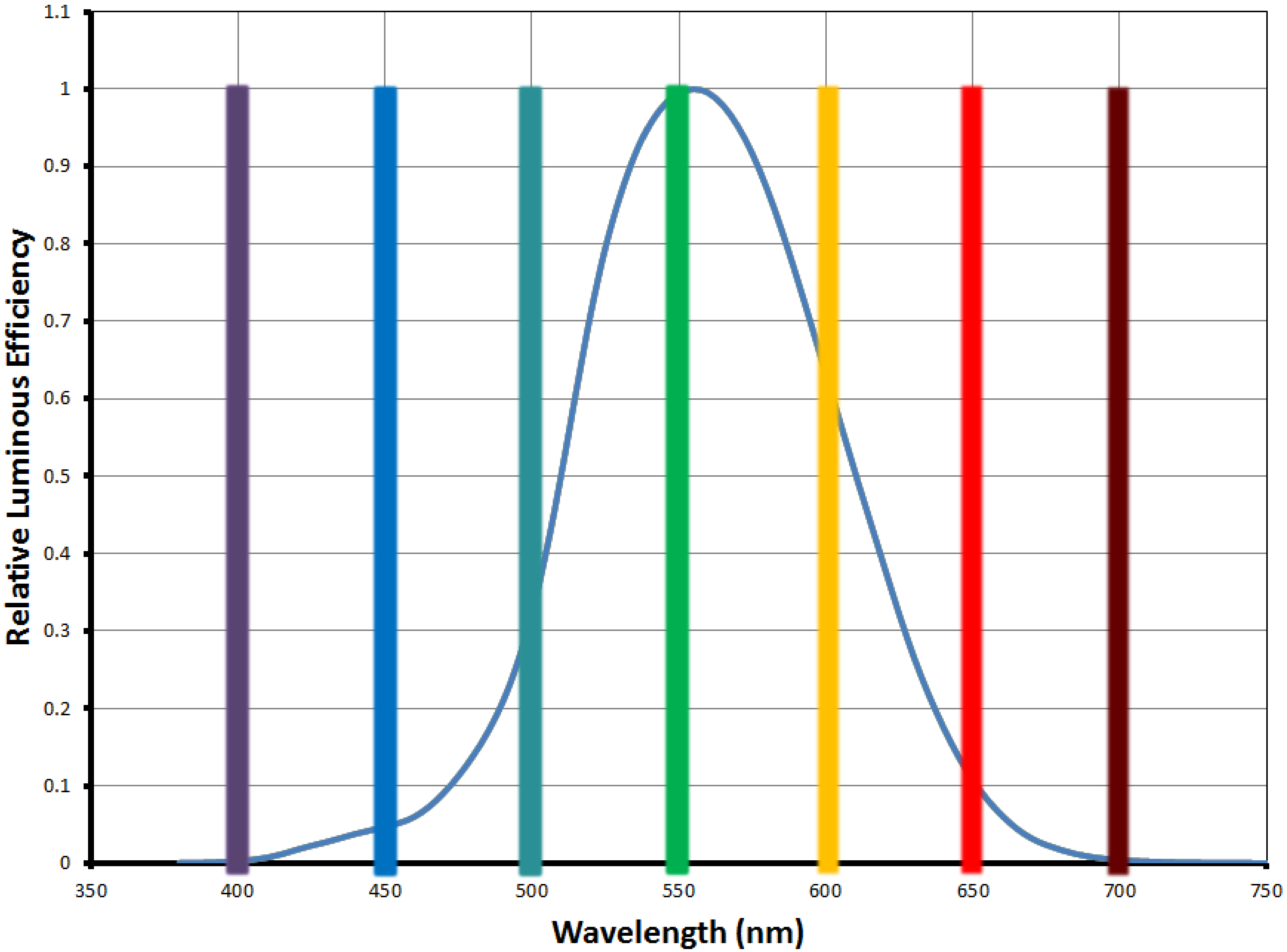
¿cuánto más intensa la interpretaría?



Deusto
Facultad de Ingeniería
Faculty of Engineering

Luminosidad y longitude de onda

Una Fuente de luz proporcinara más iluminancia (mayores valores de LUX) si concentra la energía electromagnética cerca de las longitudes de onda del verde.

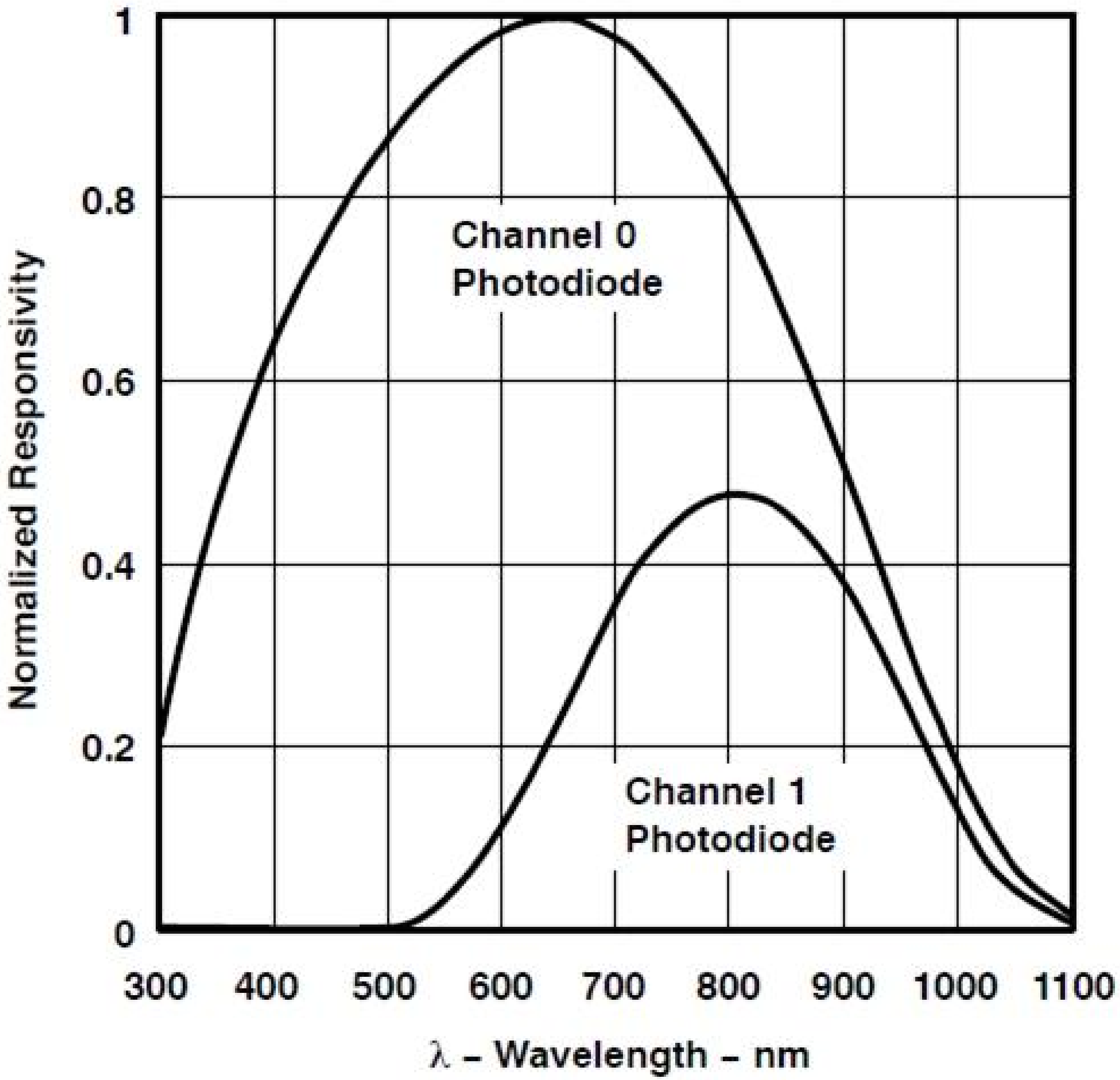


Deusto
Facultad de Ingeniería
Faculty of Engineering

Luminosidad y longitude de onda

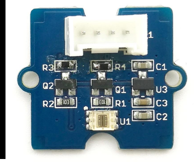
Modos de operaicón

- Espectro completo
- Sólo infrarrojo



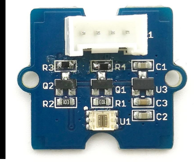
Deusto
Facultad de Ingeniería
Faculty of Engineering

Digital Light Sensor TSL2561



1. Escribimos en el registro de control para POWER ON
2. Escribimos en el registro de timing el valor de integración del ADC
3. Leemos los valores del registro de datos: Valores del sensor de luminosidad

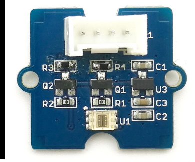
Deusto
Facultad de Ingeniería
Faculty of Engineering



TSL2561 (1)

```python
# TSL2561 address, 0x29
# Select control register, 0x00(00) with command register, 0x80(128)
#       0x03(03)    Power ON mode
bus.write_byte_data(0x29, 0x00 | 0x80, 0x03)
```

ADDRESS	RESISTER NAME	REGISTER FUNCTION
--	COMMAND	Specifies register address
0h	CONTROL	Control of basic functions
1h	TIMING	Integration time/gain control
2h	THRESHLOWLOW	Low byte of low interrupt threshold
3h	THRESHLOWHIGH	High byte of low interrupt threshold
4h	THRESHHIGHLOW	Low byte of high interrupt threshold
5h	THRESHHIGHHIGH	High byte of high interrupt threshold
6h	INTERRUPT	Interrupt control
7h	--	Reserved
8h	CRC	Factory test — not a user register
9h	--	Reserved
Ah	ID	Part number/ Rev ID
Bh	--	Reserved
Ch	DATA0LOW	Low byte of ADC channel 0
Dh	DATA0HIGH	High byte of ADC channel 0
Eh	DATA1LOW	Low byte of ADC channel 1
Fh	DATA1HIGH	High byte of ADC channel 1



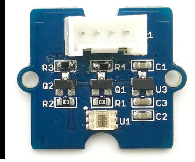
TSL2561 (1)

```python
# TSL2561 address, 0x29
# Select control register, 0x00(00) with command register, 0x80(128)
#       0x03(03)    Power ON mode
bus.write_byte_data(0x29, 0x00 | 0x80, 0x03)
```

Table 4. Control Register

	7	6	5	4	3	2	1	0	
0h	Resv	Resv	Resv	Resv	Resv	Resv	POWER		CONTROL
Reset Value:	0	0	0	0	0	0	0	0	

FIELD	BIT	DESCRIPTION
Resv	7:2	Reserved. Write as 0.
POWER	1:0	Power up/power down. By writing a 03h to this register, the device is powered up. By writing a 00h to this register, the device is powered down. ***NOTE:*** *If a value of 03h is written, the value returned during a read cycle will be 03h. This feature can be used to verify that the device is communicating properly.*



TSL2561 (2)

```
# Select timing register, 0x01(01) with command register, 0x80(128)
#       0x02(02)    Nominal integration time = 402ms
bus.write_byte_data(0x29, 0x01 | 0x80, 0x02)
```

ADDRESS	RESISTER NAME	REGISTER FUNCTION
--	COMMAND	Specifies register address
0h	CONTROL	Control of basic functions
1h	TIMING	Integration time/gain control
2h	THRESHLOWLOW	Low byte of low interrupt threshold
3h	THRESHLOWHIGH	High byte of low interrupt threshold
4h	THRESHHIGHLOW	Low byte of high interrupt threshold
5h	THRESHHIGHHIGH	High byte of high interrupt threshold
6h	INTERRUPT	Interrupt control
7h	--	Reserved
8h	CRC	Factory test — not a user register
9h	--	Reserved
Ah	ID	Part number/ Rev ID
Bh	--	Reserved
Ch	DATA0LOW	Low byte of ADC channel 0
Dh	DATA0HIGH	High byte of ADC channel 0
Eh	DATA1LOW	Low byte of ADC channel 1
Fh	DATA1HIGH	High byte of ADC channel 1

TSL2561 (2)

```
# Select timing register, 0x01(01) with command register, 0x80(128)
#       0x02(02)    Nominal integration time = 402ms
bus.write_byte_data(0x29, 0x01 | 0x80, 0x02)
```

Table 6. Integration Time

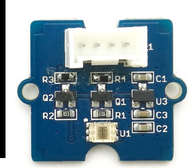
INTEG FIELD VALUE	SCALE	NOMINAL INTEGRATION TIME
00	0.034	13.7 ms
01	0.252	101 ms
10	1	402 ms
11	--	N/A

Table 5. Timing Register

	7	6	5	4	3	2	1	0	
1h	Resv	Resv	Resv	GAIN	Manual	Resv	INTEG		TIMING
Reset Value:	0	0	0	0	0	0	1	0	

FIELD	BIT	DESCRIPTION
Resv	7–5	Reserved. Write as 0.
GAIN	4	Switches gain between low gain and high gain modes. Writing a 0 selects low gain (1×); writing a 1 selects high gain (16×).
Manual	3	Manual timing control. Writing a 1 begins an integration cycle. Writing a 0 stops an integration cycle. ***NOTE:*** *This field only has meaning when INTEG = 11. It is ignored at all other times.*
Resv	2	Reserved. Write as 0.
INTEG	1:0	Integrate time. This field selects the integration time for each conversion.

Deusto
Facultad de Ingeniería
Faculty of Engineering

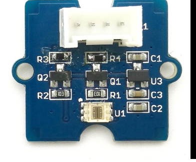


¿Cómo leemos los LUX del sensor?

ADDRESS	RESISTER NAME	REGISTER FUNCTION
--	COMMAND	Specifies register address
0h	CONTROL	Control of basic functions
1h	TIMING	Integration time/gain control
2h	THRESHLOWLOW	Low byte of low interrupt threshold
3h	THRESHLOWHIGH	High byte of low interrupt threshold
4h	THRESHHIGHLOW	Low byte of high interrupt threshold
5h	THRESHHIGHHIGH	High byte of high interrupt threshold
6h	INTERRUPT	Interrupt control
7h	--	Reserved
8h	CRC	Factory test — not a user register
9h	--	Reserved
Ah	ID	Part number/ Rev ID
Bh	--	Reserved
Ch	DATA0LOW	Low byte of ADC channel 0
Dh	DATA0HIGH	High byte of ADC channel 0
Eh	DATA1LOW	Low byte of ADC channel 1
Fh	DATA1HIGH	High byte of ADC channel 1

Debemos leer el byte de estos 4 registros

Deusto
Facultad de Ingeniería
Faculty of Engineering

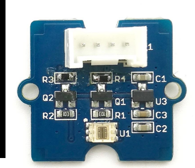


TSL2561 (3)

Ch	DATA0LOW	Low byte of ADC channel 0
Dh	DATA0HIGH	High byte of ADC channel 0
Eh	DATA1LOW	Low byte of ADC channel 1
Fh	DATA1HIGH	High byte of ADC channel 1

```
# Leer 2 bytes empezando en el registro 0x0C
with command register, 0x80(128)
# ch0 LSB, ch0 MSB
data0 = bus.read_i2c_block_data(0x29, 0x0C | 0x80, 2)

# Leer 2 bytes empezando en el registro 0x0E
with command register, 0x80(128)
# ch1 LSB, ch1 MSB
data1 = bus.read_i2c_block_data(0x29, 0x0E | 0x80, 2)
```



TSL2561 (3)

```
# Convertimos el valor leído en un valor entero
ch0 = data0[1] * 256 + data0[0] #shift dataHigh to uppber byte
ch1 = data1[1] * 256 + data1[0] #shift dataHigh to upper byte

# Espectro completo: ch0
# Rango infrarrojo: ch1
# Rango visible: ch0-ch1
```

Acelerómetro - ¿Qué mide?

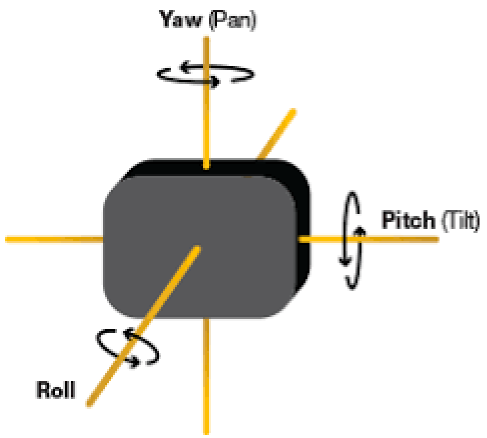
Aceleración: metro por segundo al cuadrado (m/s^2). Un cuerpo con una **aceleración** de 1 m/s^2 varía su velocidad en 1 metro/segundo cada segundo.

La **aceleración estándar debida a la gravedad** (o **aceleración estándar de caída libre**), wa la aceleración gravitacional de un objeto en el vacío cercano a la superficie de la Tierra. Está definida por estándar como 9,80665 m/s^2

1 g = 9,80665 m/s^2

- ❑ Cuando está en caída libre, ¿cómo es el modulo de (x,y,z)?

Acelerómetro: Aplicaciones



- ❑ **Pan / Tilt / Roll (camera)**
- ❑ **Vibration / "Rough-road" detection**
  - Isolating vibration of mechanical system from outside sources
- ❑ **Vehicle skid detection**
  - Deploying "smart" braking to regain control of vehicle
- ❑ **Impact detection**
  - Detecting and logging impact, when it occurs (avoid hard drive damage in laptops)
- ❑ **Input / feedback for active suspension control systems**
  - Keeps vehicle level

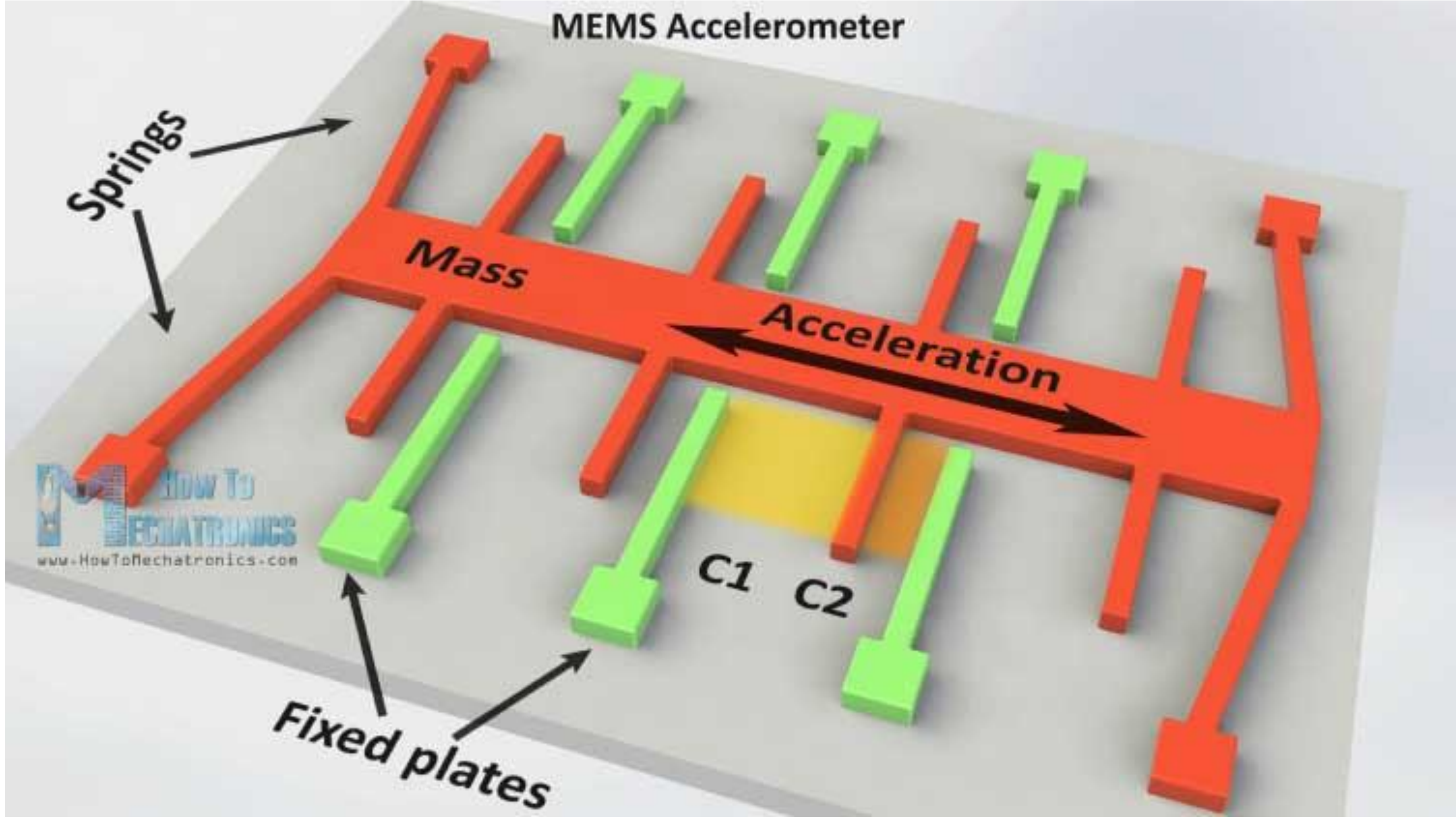
Acelerómetro MEMS capacitivo

- Mide la acelración mediante un cambio de capacidad

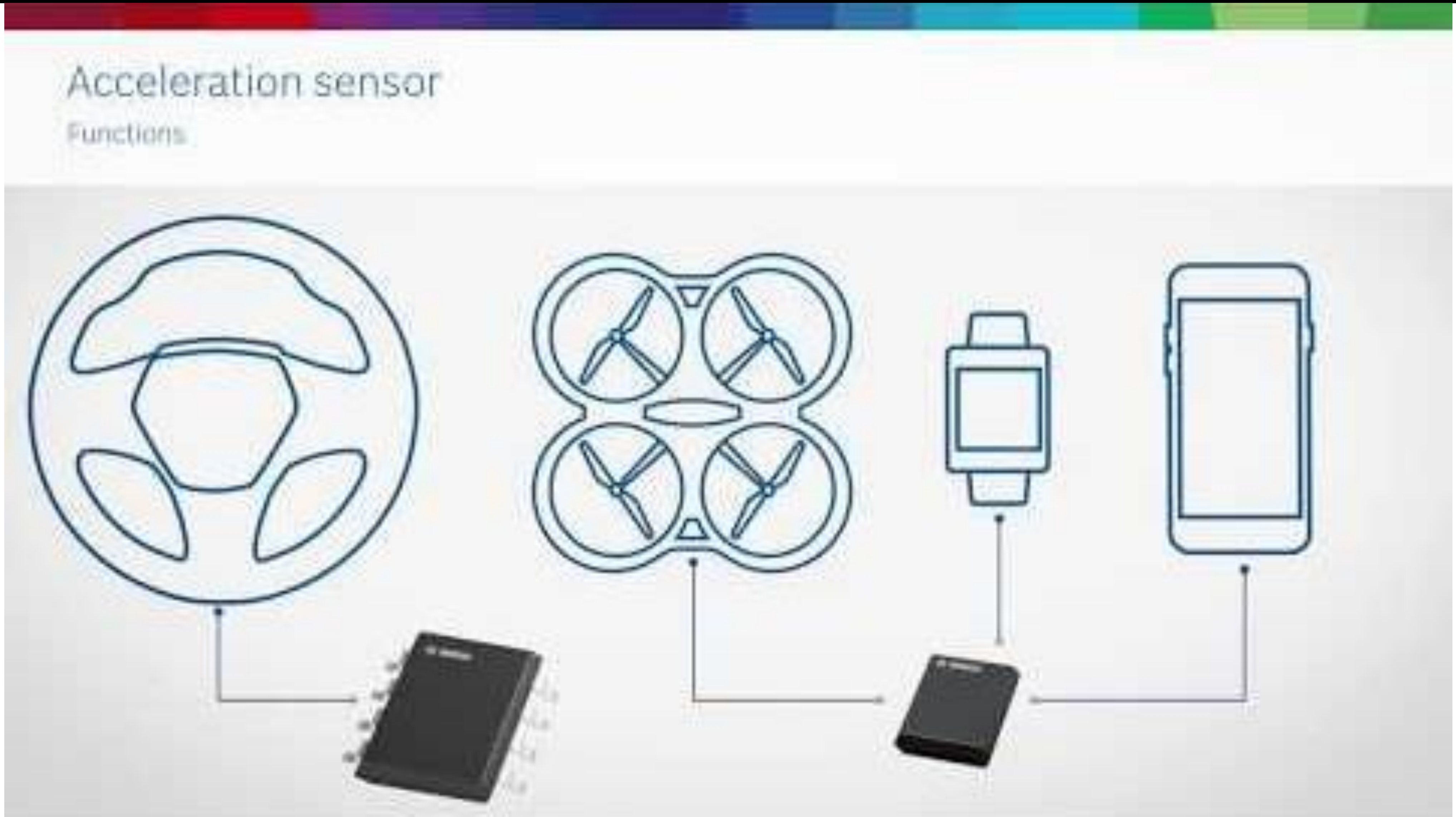
- Tiene una masa unida a un muelle que está confinado a moverse a lo largo de una dirección y unas placas

- Cuando se aplica aceleración en una dirección, la masa se moverá, y cambiará la capacidad entre la masa y el muelle

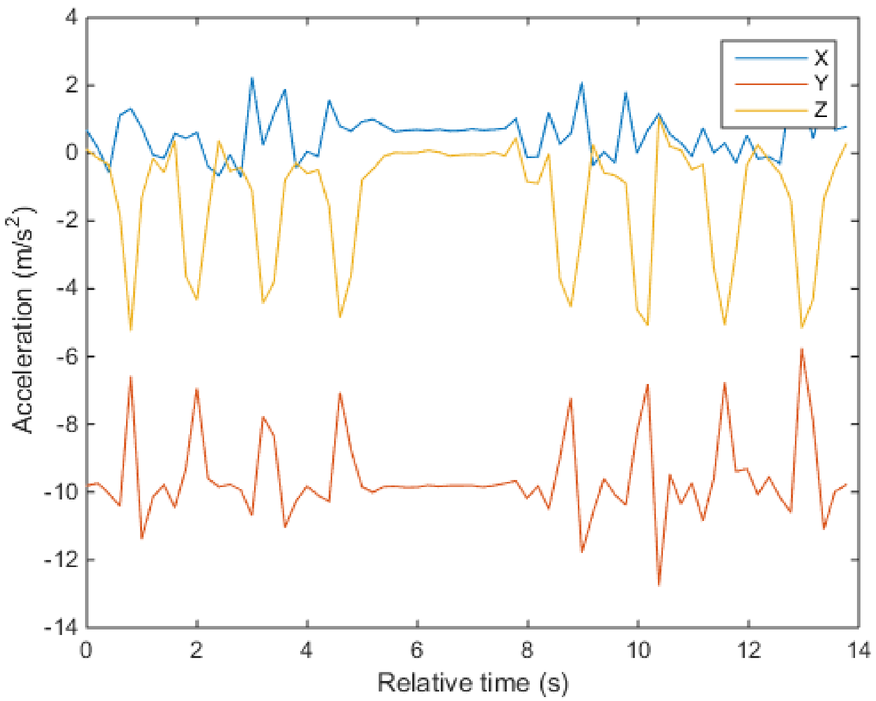
- Este cambio se puede medir, y convertir a valor de aceleración

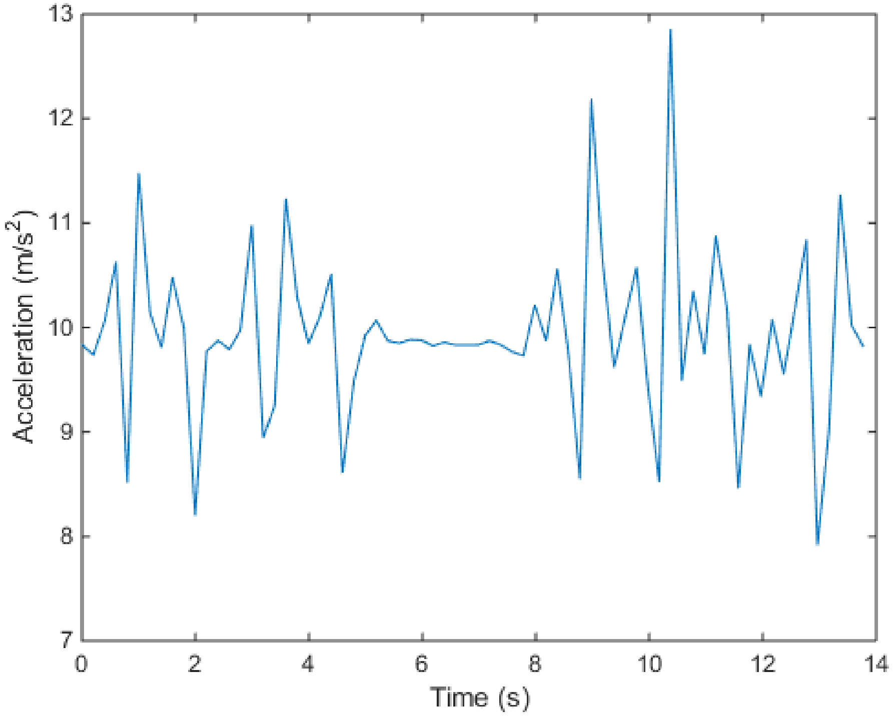


Acelerómetro: MEMs capacitivo



Acelerómetro: Contador de pasos

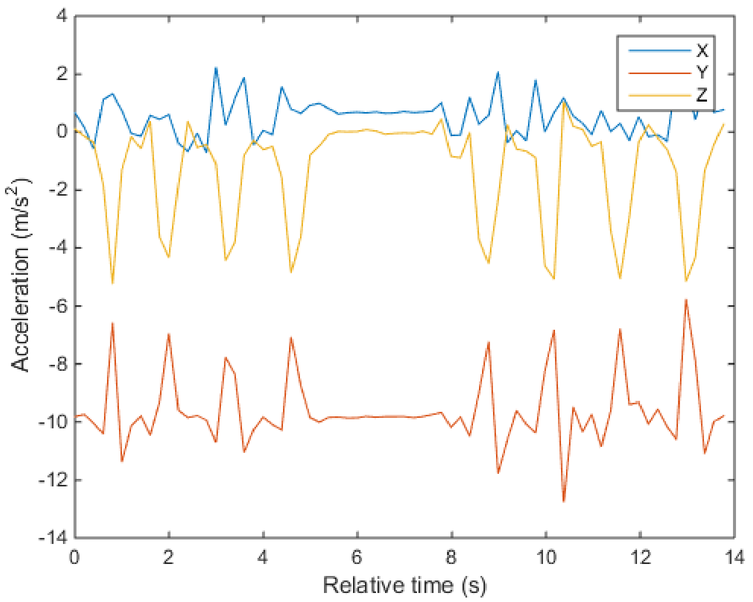




¿Cómo Podemos medir los pasos?

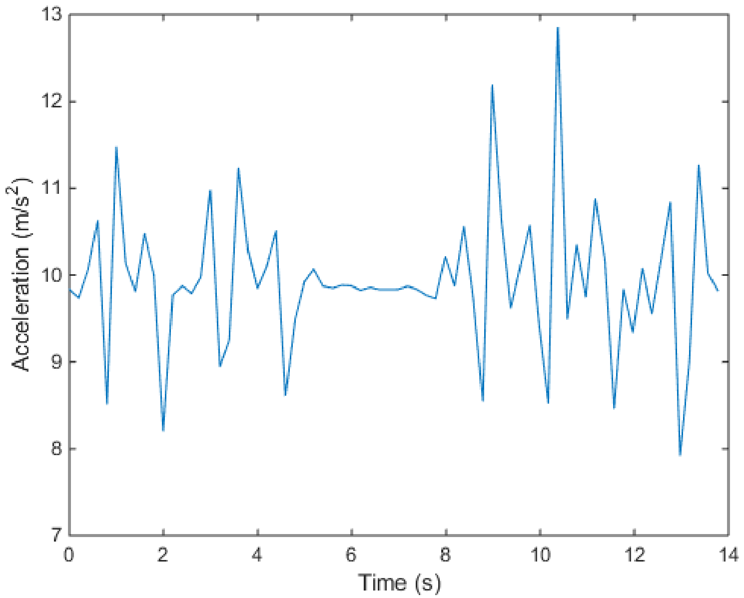
Acelerómetro: Contador de pasos

Raw data

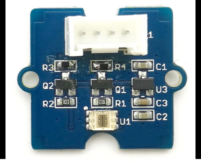


Magnitude

```
mag = sqrt(sum(x.^2 + y.^2 + z.^2, 2))
```



¿Cómo Podemos medir los pasos?



Acelerómetro MMA7660FC

1. Escribimos en el registro de SR para seleccionar el sampling rate (velocidad de muestreo)

2. Configuramos otros parámetros (ver Código de ejemplo del laboratorio 5)

3. Leemos los valores del registro de datos: Valores de aceleración

Deusto
Facultad de Ingeniería
Faculty of Engineering

MMA7660FC : configuración sample rate

REGISTER DEFINITIONS

Table 9. User Register Summary

Address	Name	Definition	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00	XOUT	6-bit output value X	-	Alert	XOUT[5]	XOUT[4]	XOUT[3]	XOUT[2]	XOUT[1]	XOUT[0]
$01	YOUT	6-bit output value Y	-	Alert	YOUT[5]	YOUT[4]	YOUT[3]	YOUT[2]	YOUT[1]	YOUT[0]
$02	ZOUT	6-bit output value Z	-	Alert	ZOUT[5]	ZOUT[4]	ZOUT[3]	ZOUT[2]	ZOUT[1]	ZOUT[0]
$03	TILT	Tilt Status	Shake	Alert	Tap	PoLa[2]	PoLa[1]	PoLa[0]	BaFro[1]	BaFro[0]
$04	SRST	Sampling Rate Status	0	0	0	0	0	0	AWSRS	AMSRS
$05	SPCNT	Sleep Count	SC[7]	SC[6]	SC[5]	SC[4]	SC[3]	SC[2]	SC[1]	SC[0]
$06	INTSU	Interrupt Setup	SHINTX	SHINTY	SHINTZ	GINT	ASINT	PDINT	PLINT	FBINT
$07	MODE	Mode	IAH	IPP	SCPS	ASE	AWE	TON	-	MODE
$08	SR	Auto-Wake/Sleep and Portrait/Landscape samples per seconds and Debounce Filter	FILT[2]	FILT[1]	FILT[0]	AWSR[1]	AWSR[0]	AMSR[2]	AMSR[1]	AMSR[0]
$09	PDET	Tap Detection	ZDA	YDA	XDA	PDTH[4]	PDTH[3]	PDTH[2]	PDTH[1]	PDTH[0]
$0A	PD	Tap Debounce Count	PD[7]	PD[6]	PD[5]	PD[4]	PD[3]	PD[2]	PD[1]	PD[0]
$0B-$1F	Factory	Reserved	-	-	-	-	-	-	-	-

MMA7660FC : configuración sample rate

```
#Función para configurar el número de muestras por segundo
   #0x4C:  Dirección del dispositivo I2C
   #0x080: dirección del registro donde queremos escribir
   #SAMPLE_RATE: valor que queremos escribir

    def sample_rate_config(self):
        SAMPLE_RATE = 0x06
        bus.write_byte_data(0x4C, 0x08, SAMPLE_RATE)
```

Acel. MMA7660FC : configuración sample rate

AMSR[2:0]	NAME	DESCRIPTION
000	AMPD	Tap Detection Mode and 120 Samples/Second Active and Auto-Sleep Mode **Tap Detection Sampling Rate**: The device takes readings continually at a rate of nominally 3846 g-cell measurements a second. It then filters these high speed measurements by maintaining continuous rolling averages of the current and last g-cell measurements. The averages are updated every 260 μs to track fast moving accelerations. Tap detection: itself compares the two filtered axis responses (fast and slow) described above for each axis. The absolute (unsigned) difference between the fast and slow axis responses is compared against the tap detection delta threshold value PDTH[4:0] in the PDET (0x09) register. **For portrait/landscape detection:** The device takes and averages 32 g-cell measurements every 8.36 ms in Active Mode and Auto-Sleep. The update rate is 120 samples per second. These measurements update the XOUT (0x00), YOUT (0x01), and ZOUT (0x02) registers also.
001	AM64	64 Samples/Second Active and Auto-Sleep Mode **For portrait/landscape detection**: The device takes and averages 32 g-cell measurements every 15.625 ms in Active Mode and Auto-Sleep. The update rate is 64 samples per second. These measurements update the XOUT (0x00), YOUT (0x01), and ZOUT (0x02) registers also.
010	AM32	32 Samples/Second Active and Auto-Sleep Mode **For portrait/landscape detection:** The device takes and averages 32 g-cell measurements every 31.25 ms in Active Mode and Auto-Sleep. The update rate is 32 samples per second. These measurements update XOUT (0x00), YOUT (0x01), and ZOUT (0x02) registers also.
011	AM16	16 Samples/Second Active and Auto-Sleep Mode **For portrait/landscape detection:** The device takes and averages 32 g-cell measurements every 62.5 ms in Active Mode and Auto-Sleep. The update rate is 16 samples per second. These measurements update the XOUT (0x00), YOUT (0x01), and ZOUT (0x02) registers also.
100	AM8	8 Samples/Second Active and Auto-Sleep Mode **For portrait/landscape detection:** The device takes and averages 32 g-cell measurements every 125 ms in Active Mode and Auto-Sleep. The update rate is 8 samples per second. These measurements update the XOUT (0x00), YOUT (0x01), and ZOUT (0x02) registers also.
101	AM4	4 Samples/Second Active and Auto-Sleep Mode **For portrait/landscape detection:** The device takes and averages 32 g-cell measurements every 250 ms in Active Mode and Auto-Sleep. The update rate is 4 samples per second. These measurements update the XOUT (0x00), YOUT (0x01), and ZOUT (0x02) registers also.
110	AM2	2 Samples/Second Active and Auto-Sleep Mode **For portrait/landscape detection:** The device takes and averages 32 g-cell measurements every 500 ms in Active Mode and Auto-Sleep. The update rate is 2 samples per second. These measurements update the XOUT (0x00), YOUT (0x01), and ZOUT (0x02) registers also.
111	AM1	1 Sample/Second Active and Auto-Sleep Mode **For portrait/landscape detection:** The device takes and averages 32 g-cell measurements every 1000 ms in Active Mode and Auto-Sleep. The update rate is 1 sample per second. These measurements update the XOUT (0x00), YOUT (0x01), and ZOUT (0x02) registers also.

MMA7660FC: Lectura de X, Y, Z

```
#Parámetros de la función bus.read_i2c_block_data
      #   0x4C : dirección I2C del dispositivo (MMA7660FC_DEFAULT_ADDRESS)
      #   0x00 donde se comienza a leer (MMA7660FC_XOUT)
      #   3: cuantos bytes se quieren leer
```

```
data = bus.read_i2c_block_data(0x4C, 0x00, 3)
```

REGISTER DEFINITIONS

Table 9. User Register Summary

Address	Name	Definition	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00	XOUT	6-bit output value X	-	Alert	XOUT[5]	XOUT[4]	XOUT[3]	XOUT[2]	XOUT[1]	XOUT[0]
$01	YOUT	6-bit output value Y	-	Alert	YOUT[5]	YOUT[4]	YOUT[3]	YOUT[2]	YOUT[1]	YOUT[0]
$02	ZOUT	6-bit output value Z	-	Alert	ZOUT[5]	ZOUT[4]	ZOUT[3]	ZOUT[2]	ZOUT[1]	ZOUT[0]
$03	TILT	Tilt Status	Shake	Alert	Tap	PoLa[2]	PoLa[1]	PoLa[0]	BaFro[1]	BaFro[0]
$04	SRST	Sampling Rate Status	0	0	0	0	0	0	AWSRS	AMSRS
$05	SPCNT	Sleep Count	SC[7]	SC[6]	SC[5]	SC[4]	SC[3]	SC[2]	SC[1]	SC[0]
$06	INTSU	Interrupt Setup	SHINTX	SHINTY	SHINTZ	GINT	ASINT	PDINT	PLINT	FBINT
$07	MODE	Mode	IAH	IPP	SCPS	ASE	AWE	TON	-	MODE
$08	SR	Auto-Wake/Sleep and Portrait/Landscape samples per seconds and Debounce Filter	FILT[2]	FILT[1]	FILT[0]	AWSR[1]	AWSR[0]	AMSR[2]	AMSR[1]	AMSR[0]
$09	PDET	Tap Detection	ZDA	YDA	XDA	PDTH[4]	PDTH[3]	PDTH[2]	PDTH[1]	PDTH[0]
$0A	PD	Tap Debounce Count	PD[7]	PD[6]	PD[5]	PD[4]	PD[3]	PD[2]	PD[1]	PD[0]
$0B-$1F	Factory	Reserved	-	-	-	-	-	-	-	-

Formato de los registros de X,Y,Z

$00: 6-bits output value X (Read Only when not in Test Mode)
XOUT — X Output

D7	D6	D5	D4	D3	D2	D1	D0
-	Alert	XOUT[5]	XOUT[4]	XOUT[3]	XOUT[2]	XOUT[1]	XOUT[0]
0	0	0	0	0	0	0	0

```
xAccl = data[0] & 0x3F
```

```
(igual ocurre para los registros de Y,Z)
```

MMA7660FC: Lectura de X, Y, Z

```python
#Función para leer los valores de x,y,z del sensor
    #Parámetros de la función bus.read_i2c_block_data
        #  dirección I2C del dispositivo (MMA7660FC_DEFAULT_ADDRESS)
        #  donde se comienza a leer (MMA7660FC_XOUT)
        #  cuantos bytes se quieren leer (3)
    def read_accl(self):
        data = bus.read_i2c_block_data(MMA7660FC_DEFAULT_ADDRESS, MMA7660FC_XOUT, 3)

        # Se convierten los datos a 6 bits, porque la trama leída completa son 8 bits, y los dos primeros
        # bits más significativos son de configuración. Los bits de la aceleración son los bits 6-0
        xAccl = data[0] & 0x3F
        if xAccl > 31 :
            xAccl -= 64

        yAccl = data[1] & 0x3F
        if yAccl > 31 :
            yAccl -= 64

        zAccl = data[2] & 0x3F
        if zAccl > 31 :
            zAccl -= 64

        return {'x' : xAccl, 'y' : yAccl, 'z' : zAccl}
```

Deusto
Facultad de Ingeniería
Faculty of Engineering

Accel: Detección de la orientación

```python
def read_orientation(self):
    ""“Leer datos del registro 0x03, 1 byte """
    data = bus.read_i2c_block_data(MMA7660FC_DEFAULT_ADDRESS, MMA7660FC_TILT, 1)
    PoLa = data & 0x1C
```

REGISTER DEFINITIONS

Table 9. User Register Summary

Address	Name	Definition	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$00	XOUT	6-bit output value X	-	Alert	XOUT[5]	XOUT[4]	XOUT[3]	XOUT[2]	XOUT[1]	XOUT[0]
$01	YOUT	6-bit output value Y	-	Alert	YOUT[5]	YOUT[4]	YOUT[3]	YOUT[2]	YOUT[1]	YOUT[0]
$02	ZOUT	6-bit output value Z	-	Alert	ZOUT[5]	ZOUT[4]	ZOUT[3]	ZOUT[2]	ZOUT[1]	ZOUT[0]
$03	TILT	Tilt Status	Shake	Alert	Tap	PoLa[2]	PoLa[1]	PoLa[0]	BaFro[1]	BaFro[0]
$04	SRST	Sampling Rate Status	0	0	0	0	0	0	AWSRS	AMSRS
$05	SPCNT	Sleep Count	SC[7]	SC[6]	SC[5]	SC[4]	SC[3]	SC[2]	SC[1]	SC[0]
$06	INTSU	Interrupt Setup	SHINTX	SHINTY	SHINTZ	GINT	ASINT	PDINT	PLINT	FBINT
$07	MODE	Mode	IAH	IPP	SCPS	ASE	AWE	TON	-	MODE
$08	SR	Auto-Wake/Sleep and Portrait/Landscape samples per seconds and Debounce Filter	FILT[2]	FILT[1]	FILT[0]	AWSR[1]	AWSR[0]	AMSR[2]	AMSR[1]	AMSR[0]
$09	PDET	Tap Detection	ZDA	YDA	XDA	PDTH[4]	PDTH[3]	PDTH[2]	PDTH[1]	PDTH[0]
$0A	PD	Tap Debounce Count	PD[7]	PD[6]	PD[5]	PD[4]	PD[3]	PD[2]	PD[1]	PD[0]
$0B-$1F	Factory	Reserved	-	-	-	-	-	-	-	-

Deusto
Facultad de Ingeniería
Faculty of Engineering

Accel: Detección de la orientación

```
if PoLa == 0x00:
    orientation = 0 #Desconocido
elif PoLa == 0x04:
    orientation = 1  #Izquierda: Dispositivo esta en modo paisaje hacia la izquierda
elif PoLa == 0x08:
    orientation = 2  #Izquierda: Dispositivo esta en modo paisaje hacia la derecha
elif PoLa == 0x14:
    orientation = 3 # Abajo: Dispositivo está en posición vertical invertida
elif PoLa == 0x18:
    orientation = 4 # Arriba: Dispositivo está en posición vertical normal
else:
    orientation = 0x00
return orientation
```

PoLa[2:0]

000: Unknown condition of up or down or left or right
001: Left: Equipment is in landscape mode to the left
010: Right: Equipment is in landscape mode to the right
101: Down: Equipment standing vertically in inverted orientation
110: Up: Equipment standing vertically in normal orientation

Actividades de clase

Formas de romper la RaspberryPi

Nunca tocar la placa con las manos mientras esté alimentada: riesgo de CORTOCIRCUITO

No la desenchufes directamente, mejor cierra ordenadamente el SO (`shutdown now`)

No pongas la placa encima de superficies metálicas, así evitaremos un cortocircuito. Usa patas de goma o mejor una carcasa

No conectes circuitos que drenen o aporten mucha corriente por las GPIO, el máximo es 2 2-3 mA (en comparación Arduino permite hasta 40 mA, pudiento alimentar circuitos por las GPIO, RPi no). GPIO máximo 3.3V

Siempre repasa 2 o 3 veces la numeración de los pines GPIO: el uso de un PIN incorrecto puede quemar la placa

Mucho cuidado cuando trabajes en modo superusuario/root ya que puedes desconfigurar el sistema

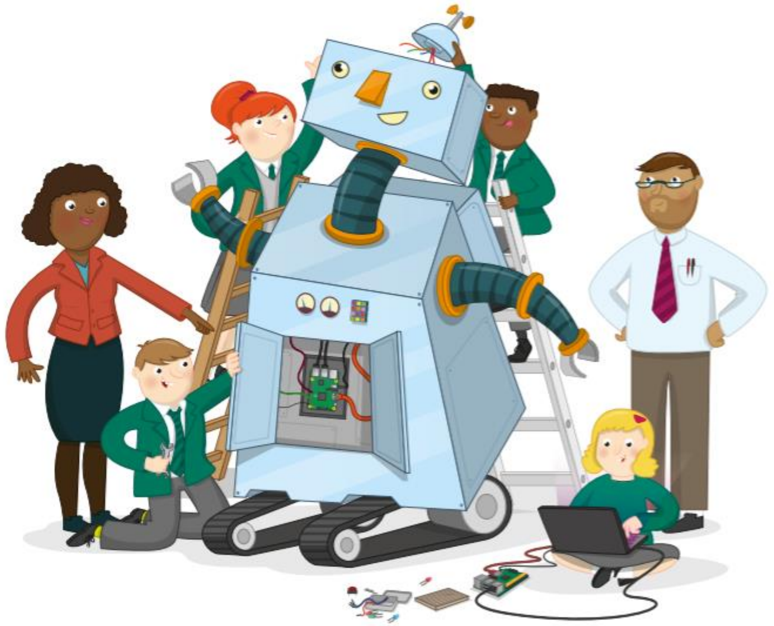


Illustration: www.raspberrypi.org

Ante cualquier duda pide ayuda al profesor o consulta Internet en más de una fuente para verificar que la solución es correcta.
Extra-cuidado: los problemas de software son reversibles, los de hardware no.

Deusto
Facultad de Ingeniería
Faculty of Engineering