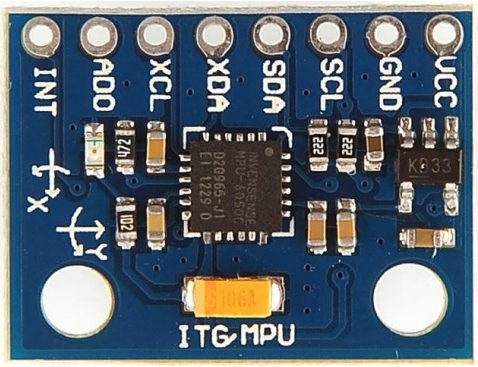
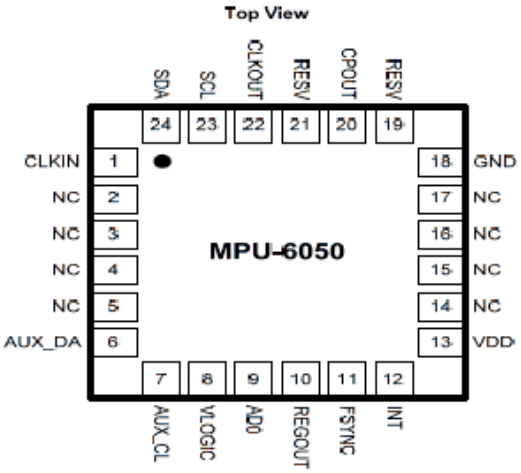
# MPU 6050

**Application Arduino avec Matlab, Simulink, StateFlow par [Marc Jakubowicz]**

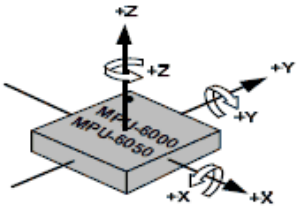
**Le 2 avril 2016 Version 1.0  Matlab 2016a**

## Mise en œuvre de l'accéléromètre gyroscope numérique MPU 6050



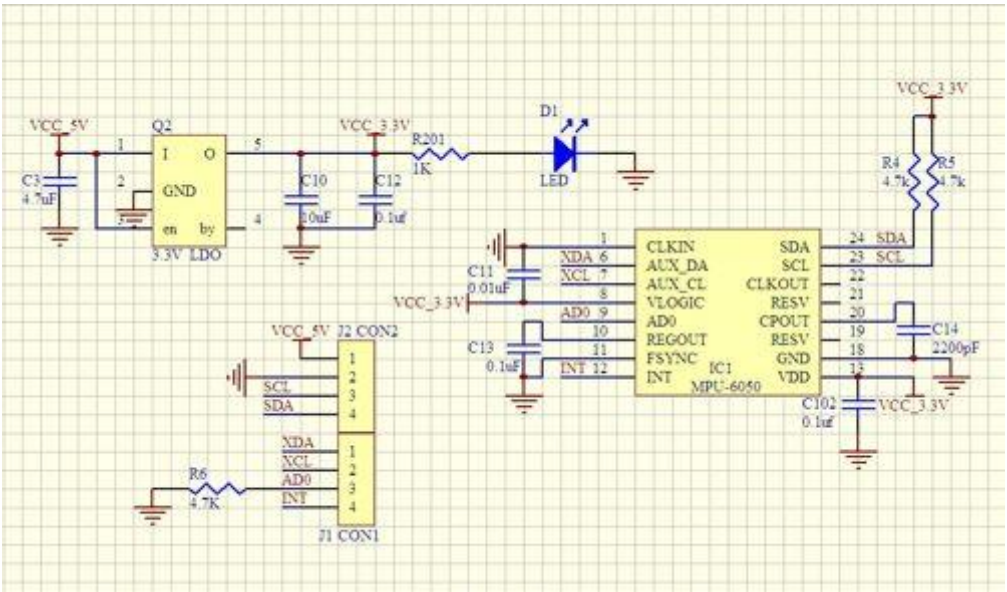




QFN Package
24-pin, 4mm x 4mm x 0.9mm



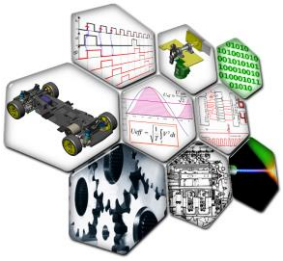

Orientation of Axes of Sensitivity and Polarity of Rotation

## Schéma structurel de la carte du capteur.



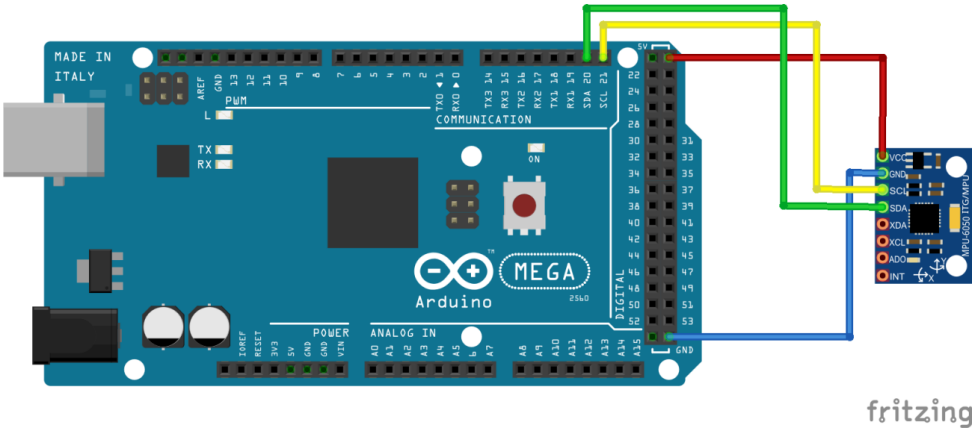

## Capteur disponible :

- http://fr.aliexpress.com/wholesale?catId=0&initiative_id=SB_20160331043101&SearchText=mpu+6050

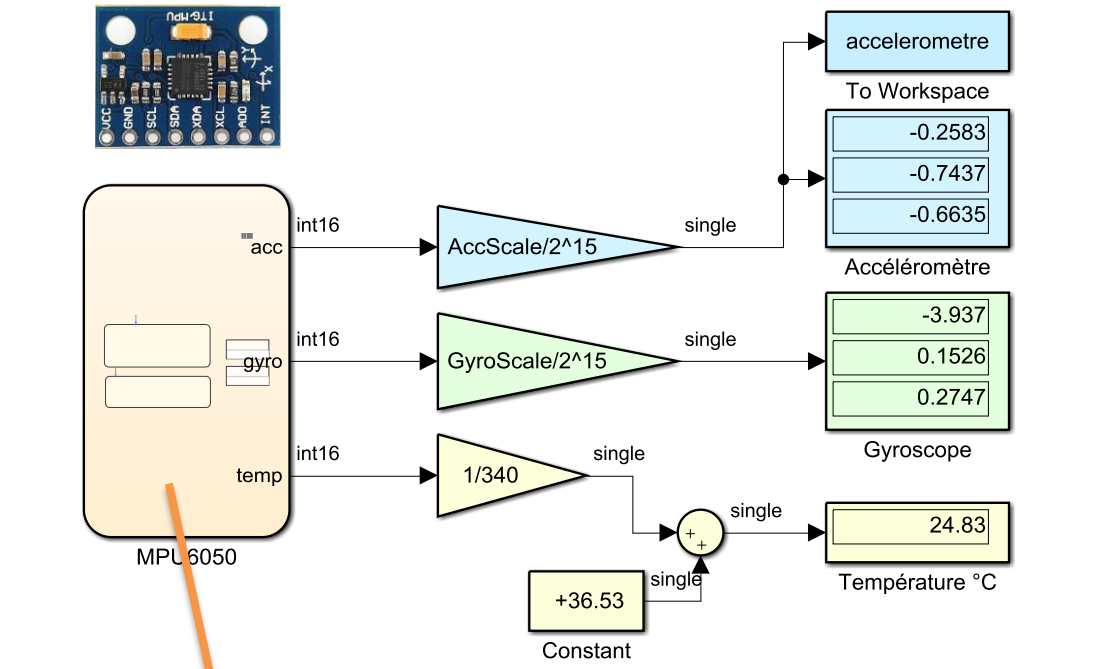


Jakubowicz Marc 2016

## Schéma de câblage avec une carte Arduino Mega 2560




## Modèle de simulation : MPU6050_acq_sf.slx

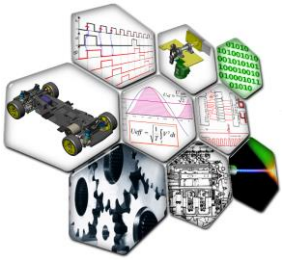



```
initialisation
en:
i2cWr(uint16(hex2dec( '6B00' )));%Power on
i2cWr(uint16(hex2dec( '1B10' )));%gyro 0x10=1000°/s
i2cWr(uint16(hex2dec( '1C08' )));%acce 0x08= 4G
i2cWr(uint16(hex2dec( '1A06' )));%filtrage+actu 0x06= 19ms
i2cWr(uint16(hex2dec( '3710' )));%INT_PIN_CFG
i2cWr(uint16(hex2dec( '3800' )));%Interrupt Enable 0 désactivé
```

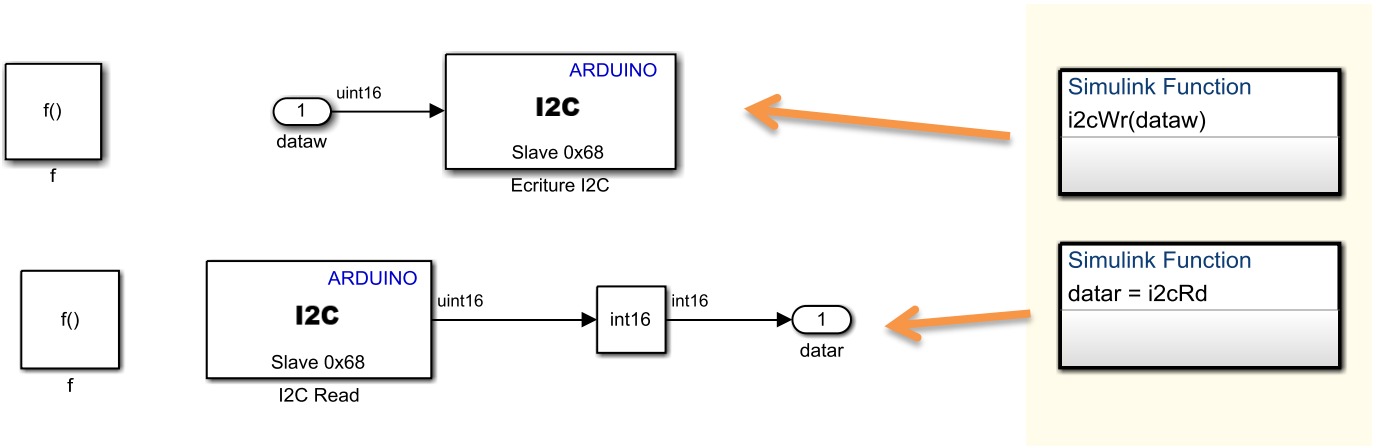
```
Lecture_acceleration3_axes
during:
val=i2cRd;
acc=val(1:3);
gyro=val(5:7);
temp=val(4:4);
```

Simulink Function
i2cWr(dataw)

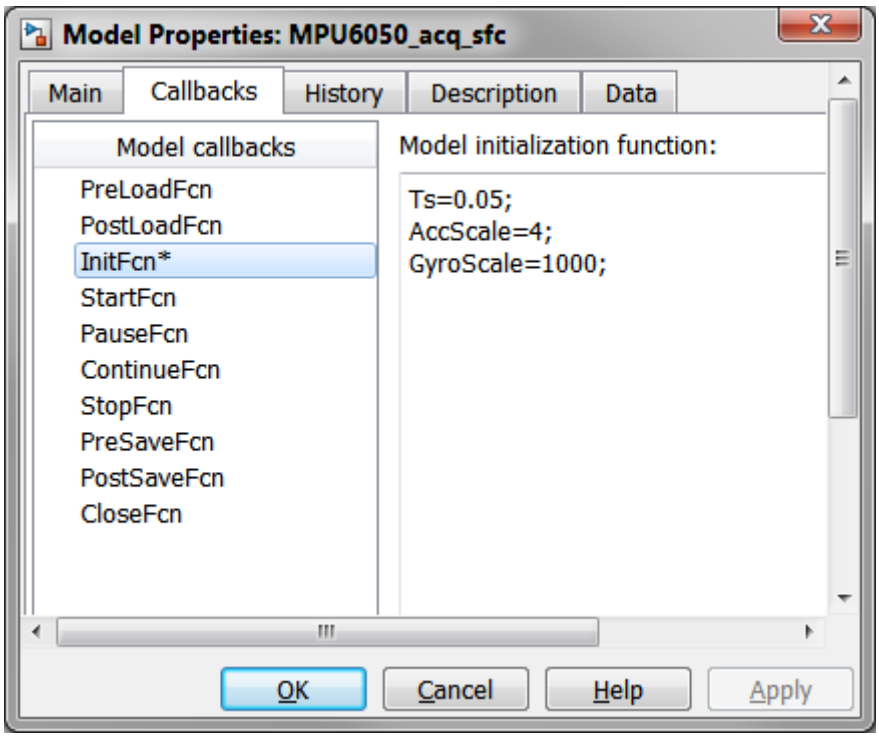
Simulink Function
datar = i2cRd

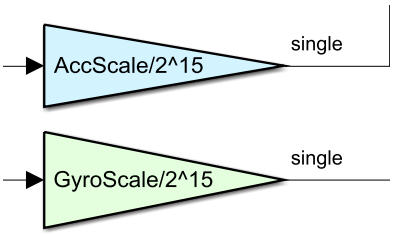


Jakubowicz Marc 2016




**Propriétés du modèle :**



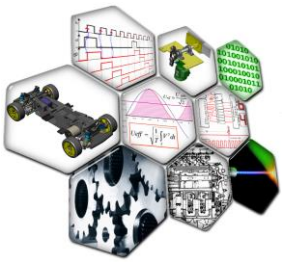




**Accélération (X Y Z) :** $-2^{15} < \mathbf{acc} < 2^{15}$

Paramètre 4G Mode 4G (**1C08**) sortie $-4 < G < 4$

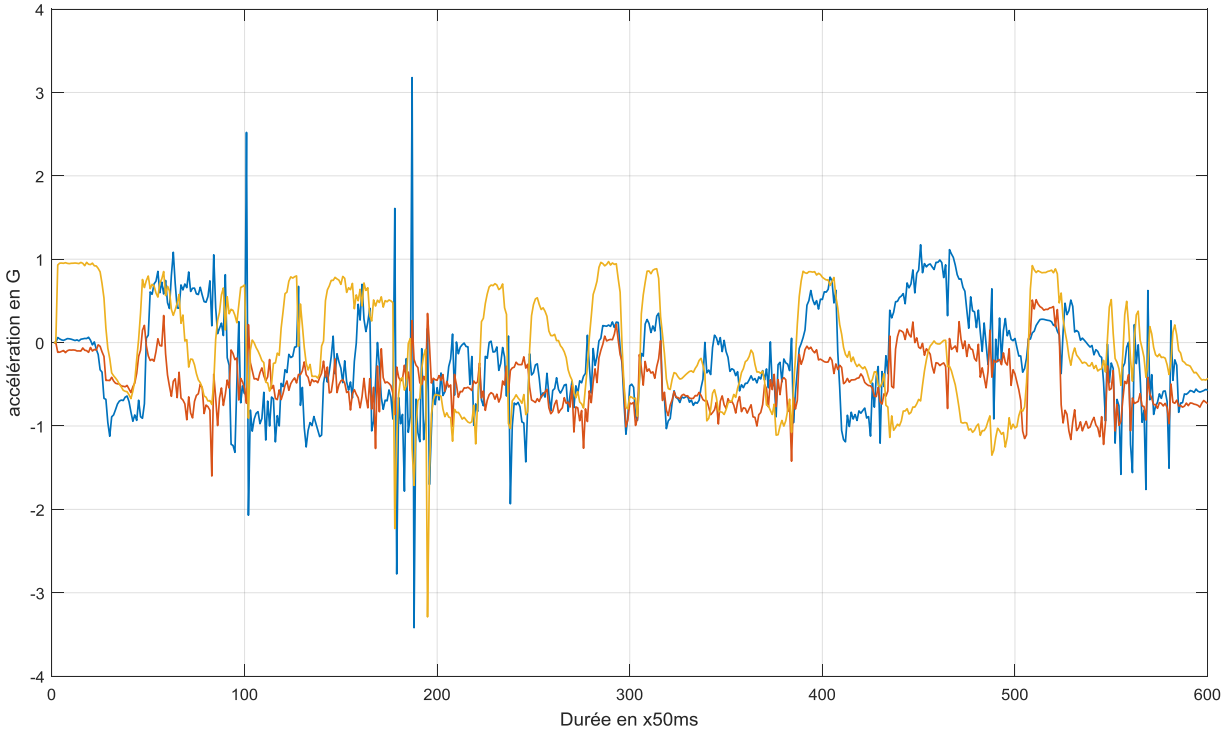
**Gyroscope (X Y Z) :** $-2^{15} < gyro < 2^{15}$

Paramètre $1000°.s^{-1}$ Mode 1000 (**1B10**) sortie $-1000 < gyro < 1000$

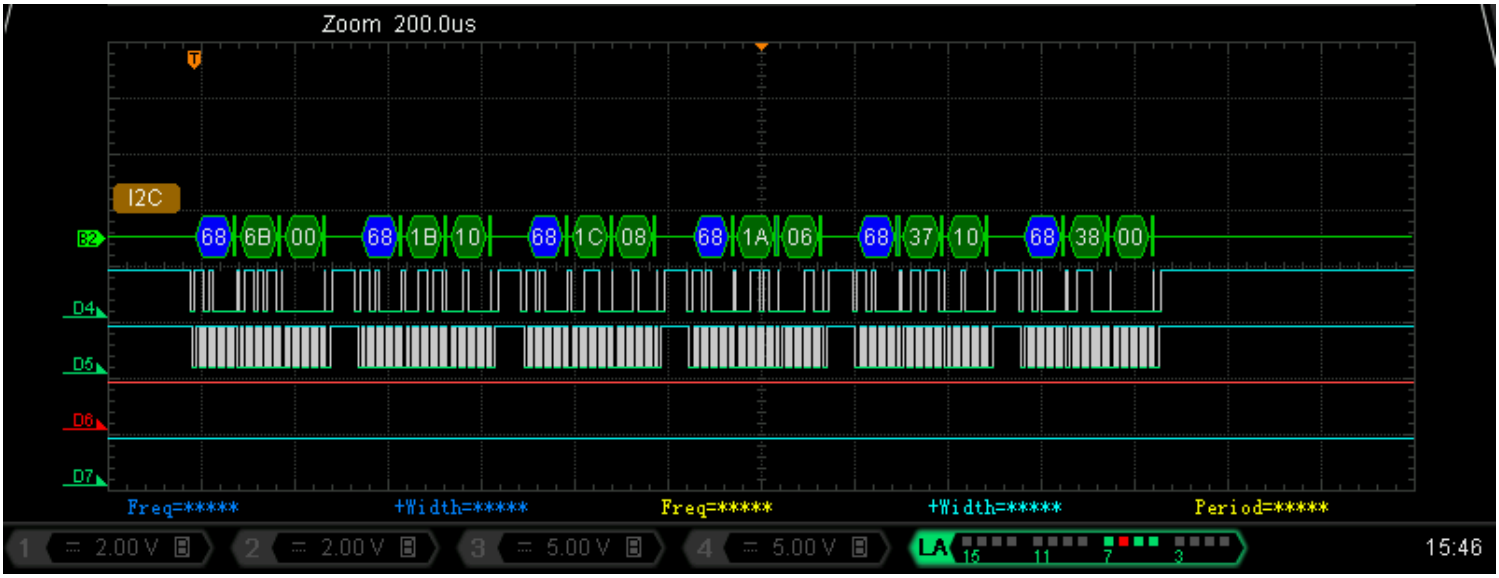


Jakubowicz Marc 2016

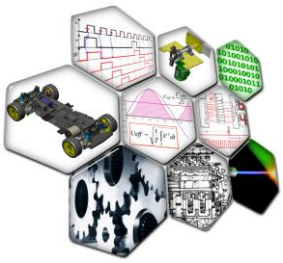
**Visualisation enregistrement "to Workspace"** *plot(accelerometre,'DisplayName','accelerometre')*




**Chronogrammes Initialisation :**

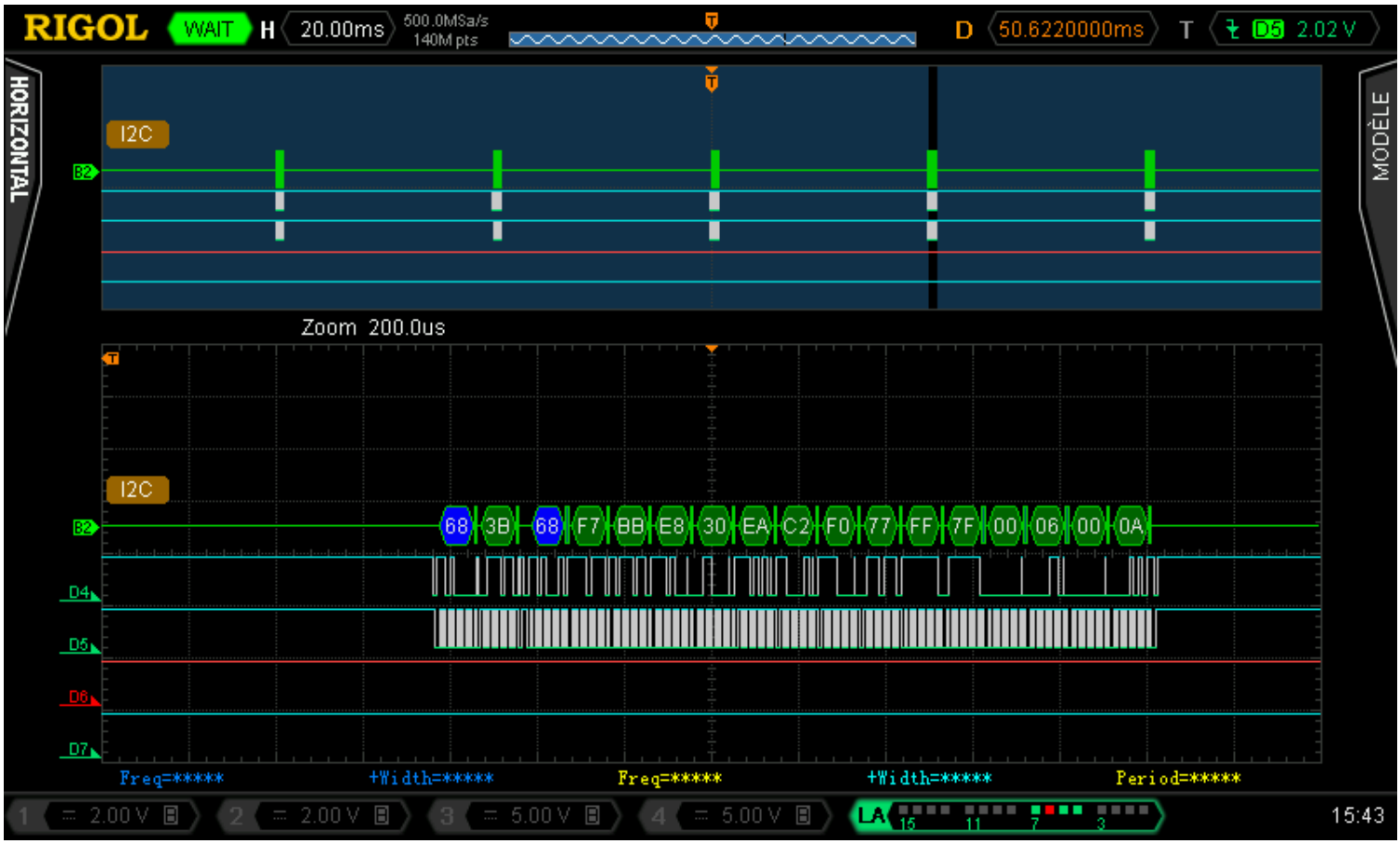



```
i2cWr(uint16(hex2dec( '6B00' ))); %Power on
i2cWr(uint16(hex2dec( '1B10' ))); %gyro 0x10=1000°/s
i2cWr(uint16(hex2dec( '1C08' ))); %acce 0x08= 4G
i2cWr(uint16(hex2dec( '1A06' ))); %filtrage+actu 0x06= 19ms
i2cWr(uint16(hex2dec( '3710' ))); %INT_PIN_CFG
i2cWr(uint16(hex2dec( '3800' ))); %Interrupt Enable 0 désactivé
```



Jakubowicz Marc 2016

## Chronogrammes mesures :




**0x68 adresse I2C du composant**

**0x3B adresse de départ de lecture des registres.**

**accXH, accXL, accYH, accYL, accZH, accZL, tempH, TempL, gyroXH, gyroXL, gyroYH, gyroYL, gyroZH, gyroZL**

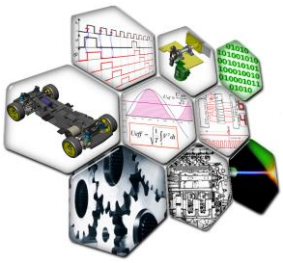
## Compilation :

## Simulation externe :

```
AVR Memory Usage
Device: atmega2560
Program: 24638 bytes (9.4% Full) (.text + .data + .bootloader)
Data: 1844 bytes (22.5% Full) (.data + .bss + .noinit)
```

## Déployé sur le hardware :

```
AVR Memory Usage
Device: atmega2560
Program: 7506 bytes (2.9% Full) (.text + .data + .bootloader)
Data: 991 bytes (12.1% Full) (.data + .bss + .noinit)
```



Jakubowicz Marc 2016

## Initialisation du composant :

# i2cWr(uint16(hex2dec( '6B00' )));

### 4.27 Register 104 – Signal Path Reset SIGNAL_PATH_RESET

**Type: Write Only**

| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 68 | 104 | - | - | - | - | - | GYRO _RESET | ACCEL _RESET | TEMP _RESET |

**Description:**

This register is used to reset the analog and digital signal paths of the gyroscope, accelerometer, and temperature sensors.

The reset will revert the signal path analog to digital converters and filters to their power up configurations.

Note: This register does not clear the sensor registers.

Bits 7 to 3 are reserved.

**Parameters:**

*GYRO_RESET* When set to 1, this bit resets the gyroscope analog and digital signal paths.

*ACCEL_RESET* When set to 1, this bit resets the accelerometer analog and digital signal paths.

*TEMP_RESET* When set to 1, this bit resets the temperature sensor analog and digital signal paths.

# i2cWr(uint16(hex2dec( '1B10' )));

### 4.4 Register 27 – Gyroscope Configuration GYRO_CONFIG

**Type: Read/Write**

| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 1B | 27 | XG_ST | YG_ST | ZG_ST | FS_SEL[1:0] | | - | - | - |

**Description:**

This register is used to trigger gyroscope self-test and configure the gyroscopes' full scale range.

Gyroscope self-test permits users to test the mechanical and electrical portions of the gyroscope. The self-test for each gyroscope axis can be activated by controlling the *XG_ST*, *YG_ST*, and *ZG_ST* bits of this register. Self-test for each axis may be performed independently or all at the same time.

When self-test is activated, the on-board electronics will actuate the appropriate sensor. This actuation will move the sensor's proof masses over a distance equivalent to a pre-defined Coriolis force. This proof mass displacement results in a change in the sensor output, which is reflected in the output signal. The output signal is used to observe the self-test response.

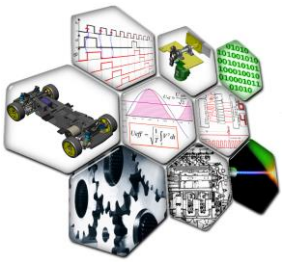
The self-test response is defined as follows:

Self-test response = Sensor output with self-test enabled – Sensor output without self-test enabled

The self-test limits for each gyroscope axis is provided in the electrical characteristics tables of the MPU-6000/MPU-6050 Product Specification document. When the value of the self-test response is within the min/max limits of the product specification, the part has passed self test. When the self-test response exceeds the min/max values specified in the document, the part is deemed to have failed self-test.

*FS_SEL* selects the full scale range of the gyroscope outputs according to the following table.

| FS_SEL | Full Scale Range |
|---|---|
| 0 | ± 250 °/s |
| 1 | ± 500 °/s |
| 2 | ± 1000 °/s |
| 3 | ± 2000 °/s |



Jakubowicz Marc 2016

# i2cWr(uint16(hex2dec( '1C08' )));

### 4.5 Register 28 – Accelerometer Configuration
### ACCEL_CONFIG

**Type: Read/Write**

| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 1C | 28 | XA_ST | YA_ST | ZA_ST | AFS_SEL[1:0] | | | - | |

**Description:**

This register is used to trigger accelerometer self test and configure the accelerometer full scale range. This register also configures the Digital High Pass Filter (DHPF).

Accelerometer self-test permits users to test the mechanical and electrical portions of the accelerometer. The self-test for each accelerometer axis can be activated by controlling the *XA_ST*, *YA_ST*, and *ZA_ST* bits of this register. Self-test for each axis may be performed independently or all at the same time.

When self-test is activated, the on-board electronics will actuate the appropriate sensor. This actuation simulates an external force. The actuated sensor, in turn, will produce a corresponding output signal. The output signal is used to observe the self-test response.

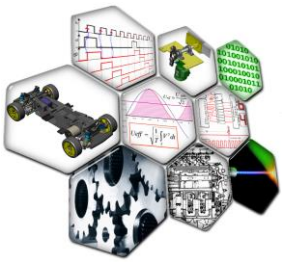
The self-test response is defined as follows:

Self-test response = Sensor output with self-test enabled – Sensor output without self-test enabled

The self-test limits for each accelerometer axis is provided in the electrical characteristics tables of the MPU-6000/MPU-6050 Product Specification document. When the value of the self-test response is within the min/max limits of the product specification, the part has passed self test. When the self-test response exceeds the min/max values specified in the document, the part is deemed to have failed self-test.

*AFS_SEL* selects the full scale range of the accelerometer outputs according to the following table.

| AFS_SEL | Full Scale Range |
|---|---|
| 0 | ± 2*g* |
| 1 | ± 4*g* |
| 2 | ± 8*g* |
| 3 | ± 16*g* |



Jakubowicz Marc 2016

# i2cWr(uint16(hex2dec( '1A06' )));  *06= 0b 0000 0110*

### 4.3 Register 26 – Configuration
**CONFIG**

**Type: Read/Write**

| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 1A | 26 | - | - | EXT_SYNC_SET[2:0] | | | DLPF_CFG[2:0] | | |

**Description:**

This register configures the external Frame Synchronization (FSYNC) pin sampling and the Digital Low Pass Filter (DLPF) setting for both the gyroscopes and accelerometers.

An external signal connected to the FSYNC pin can be sampled by configuring *EXT_SYNC_SET*.

Signal changes to the FSYNC pin are latched so that short strobes may be captured. The latched FSYNC signal will be sampled at the Sampling Rate, as defined in register 25. After sampling, the latch will reset to the current FSYNC signal state.

The sampled value will be reported in place of the least significant bit in a sensor data register determined by the value of *EXT_SYNC_SET* according to the following table.

| EXT_SYNC_SET | FSYNC Bit Location |
|---|---|
| 0 | Input disabled |
| 1 | TEMP_OUT_L[0] |
| 2 | GYRO_XOUT_L[0] |
| 3 | GYRO_YOUT_L[0] |
| 4 | GYRO_ZOUT_L[0] |
| 5 | ACCEL_XOUT_L[0] |
| 6 | ACCEL_YOUT_L[0] |
| 7 | ACCEL_ZOUT_L[0] |

The DLPF is configured by *DLPF_CFG*. The accelerometer and gyroscope are filtered according to the value of *DLPF_CFG* as shown in the table below.

| DLPF_CFG | Accelerometer ($F_s$ = 1kHz) | | Gyroscope | | |
|---|---|---|---|---|---|
| | Bandwidth (Hz) | Delay (ms) | Bandwidth (Hz) | Delay (ms) | Fs (kHz) |
| 0 | 260 | 0 | 256 | 0.98 | 8 |
| 1 | 184 | 2.0 | 188 | 1.9 | 1 |
| 2 | 94 | 3.0 | 98 | 2.8 | 1 |
| 3 | 44 | 4.9 | 42 | 4.8 | 1 |
| 4 | 21 | 8.5 | 20 | 8.3 | 1 |
| 5 | 10 | 13.8 | 10 | 13.4 | 1 |
| 6 | 5 | 19.0 | 5 | 18.6 | 1 |
| 7 | RESERVED | | RESERVED | | 8 |

# i2cWr(uint16(hex2dec( '3710' )));

### 4.15 Register 55 – INT Pin / Bypass Enable Configuration
**INT_PIN_CFG**

**Type: Read/Write**

| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 37 | 55 | INT_LEVEL | INT_OPEN | LATCH_INT_EN | INT_RD_CLEAR | FSYNC_INT_LEVEL | FSYNC_INT_EN | I2C_BYPASS_EN | - |

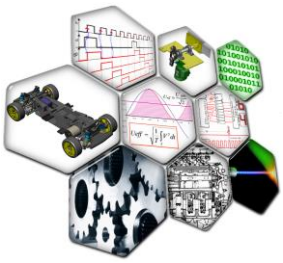
**Description:**

This register configures the behavior of the interrupt signals at the INT pins. This register is also used to enable the FSYNC Pin to be used as an interrupt to the host application processor, as well as to enable Bypass Mode on the $I^2C$ Master. This bit also enables the clock output.

*FSYNC_INT_EN* enables the FSYNC pin to be used as an interrupt to the host application processor. A transition to the active level specified in *FSYNC_INT_LEVEL* will trigger an interrupt. The status of this interrupt is read from the *PASS_THROUGH* bit in the $I^2C$ Master Status Register (Register 54).

When *I2C_BYPASS_EN* is equal to 1 and *I2C_MST_EN* (Register 106 bit[5]) is equal to 0, the host application processor will be able to directly access the auxiliary $I^2C$ bus of the MPU-60X0. When this bit is equal to 0, the host application processor will not be able to directly access the auxiliary $I^2C$ bus of the MPU-60X0 regardless of the state of *I2C_MST_EN*.

For further information regarding Bypass Mode, please refer to Section 7.11 and 7.13 of the MPU-6000/MPU-6050 Product Specification document.



Jakubowicz Marc 2016

# i2cWr(uint16(hex2dec( '3800' )));

### 4.16 Register 56 – Interrupt Enable
### INT_ENABLE

**Type: Read/Write**

| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 38 | 56 | | MOT_EN | | FIFO _OFLOW _EN | I2C_MST _INT_EN | - | - | DATA _RDY_EN |

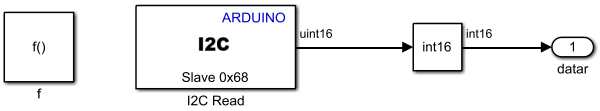
**Description:**

This register enables interrupt generation by interrupt sources.

For information regarding the interrupt status for each interrupt generation source, please refer to Register 58. Further information regarding $I^2C$ Master interrupt generation can be found in Register 54.

Bits 2 and 1 are reserved.

# Lecture des valeurs




### 4.18 Registers 59 to 64 – Accelerometer Measurements
### ACCEL_XOUT_H, ACCEL_XOUT_L, ACCEL_YOUT_H, ACCEL_YOUT_L, ACCEL_ZOUT_H, and ACCEL_ZOUT_L

**Type: Read Only**

| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 3B | 59 | ACCEL_XOUT[15:8] | | | | | | | |
| 3C | 60 | ACCEL_XOUT[7:0] | | | | | | | |
| 3D | 61 | ACCEL_YOUT[15:8] | | | | | | | |
| 3E | 62 | ACCEL_YOUT[7:0] | | | | | | | |
| 3F | 63 | ACCEL_ZOUT[15:8] | | | | | | | |
| 40 | 64 | ACCEL_ZOUT[7:0] | | | | | | | |

### 4.19 Registers 65 and 66 – Temperature Measurement
### TEMP_OUT_H and TEMP_OUT_L

**Type: Read Only**

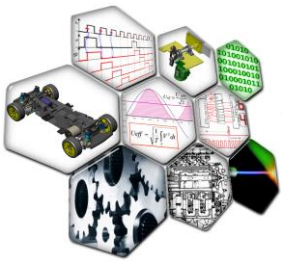
| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 41 | 65 | TEMP_OUT[15:8] | | | | | | | |
| 42 | 66 | TEMP_OUT[7:0] | | | | | | | |

Temperature in degrees C = (TEMP_OUT Register Value as a signed quantity)/340 + 36.53

### 4.20 Registers 67 to 72 – Gyroscope Measurements
### GYRO_XOUT_H, GYRO_XOUT_L, GYRO_YOUT_H, GYRO_YOUT_L, GYRO_ZOUT_H, and GYRO_ZOUT_L

**Type: Read Only**

| Register (Hex) | Register (Decimal) | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 43 | 67 | GYRO_XOUT[15:8] | | | | | | | |
| 44 | 68 | GYRO_XOUT[7:0] | | | | | | | |
| 45 | 69 | GYRO_YOUT[15:8] | | | | | | | |
| 46 | 70 | GYRO_YOUT[7:0] | | | | | | | |
| 47 | 71 | GYRO_ZOUT[15:8] | | | | | | | |
| 48 | 72 | GYRO_ZOUT[7:0] | | | | | | | |



Jakubowicz Marc 2016