Utilizarea dispozitivului MPU6050

Lucrare de labrator

Contents

[1. Introducere 3](#_Toc472212008)

[2. Conectarea dispozitivelor: 4](#_Toc472212009)

[3. Pregatirea mediului software pentru lucrare de laborator 6](#_Toc472212010)

[4. TEMA – 1 6](#_Toc472212011)

[5. TEMA – 2 9](#_Toc472212012)

# Introducere

In acesta lucrare veti face cunostinta cu chipul MPU605 care contine giroscop, accelerometru si termometru. Noi vom folosi numai accelerometru.

Vet citi si prelucra datele de la acest dispozitiv. In final veti avea un demo asemanator cu ce este prezentat in acest video:***MPU\_demo***(gasiti in folderol de starter)

Accelerometrul este conectat la Arduino prini interfata I2C. Arduino este conectat la calculator prin USART (pe arduino este convertit USART la USB). Arduino aprinde LEDurile pe placa trimitant +5V pentru LEDurile corespunzatoare.

ARDUINO

PC (PUTTY)

MPU6050

LEDuri

+5V

I2C

USART

USART si I2C sunt implementate in limbaj C.

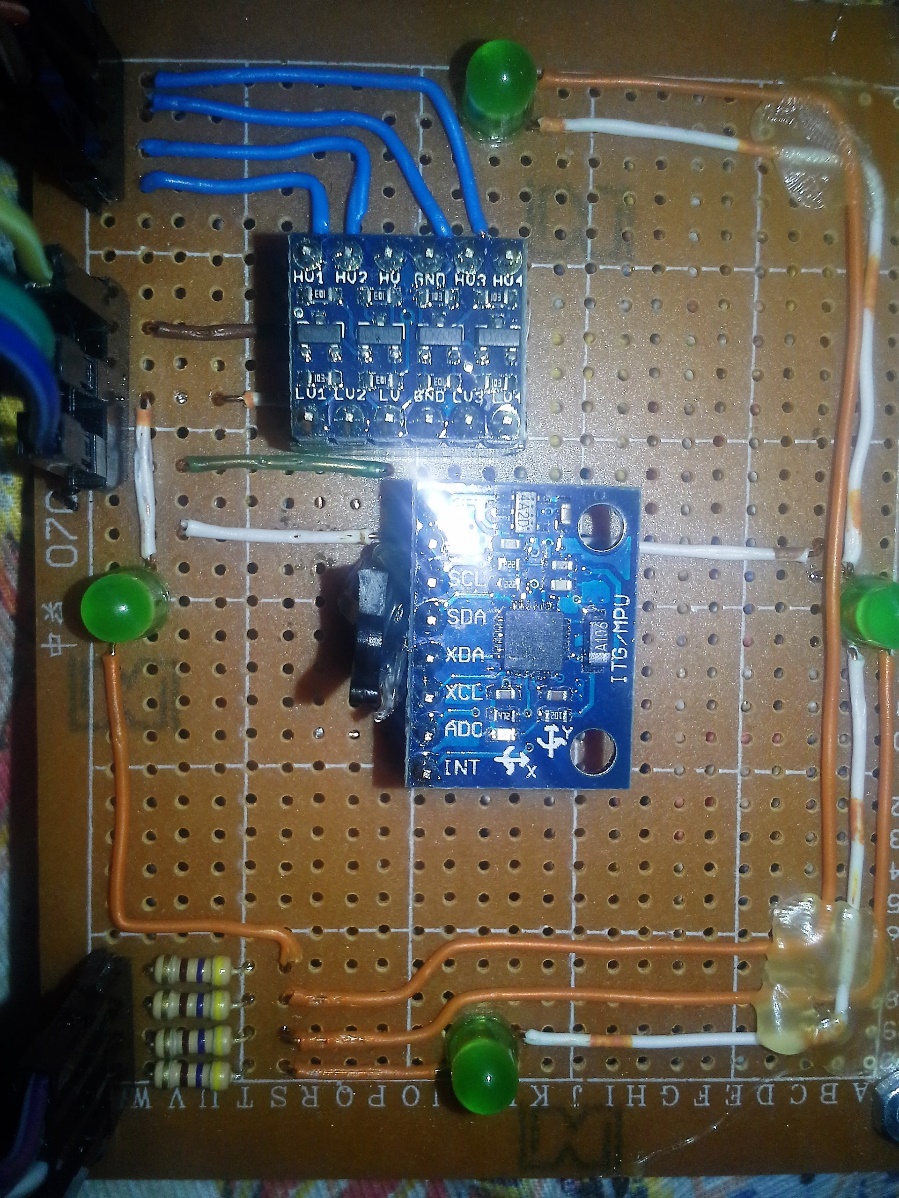
Acest laborator nu are ca obiectiv descrierea protocoalelor si a implementarii acestora in C.

# Conectarea dispozitivelor:

1. Deschideti fisierul *MPU\_demo.mp4*. Asta va rezulta aceasta lucrare. Sa incepem!
2. Montati translatorul de nivel in soclu (atentie la nivele! )
3. Montai MPU6050 in soclu

**2**

**1**



**HV (5v)**

**LV (3v)**

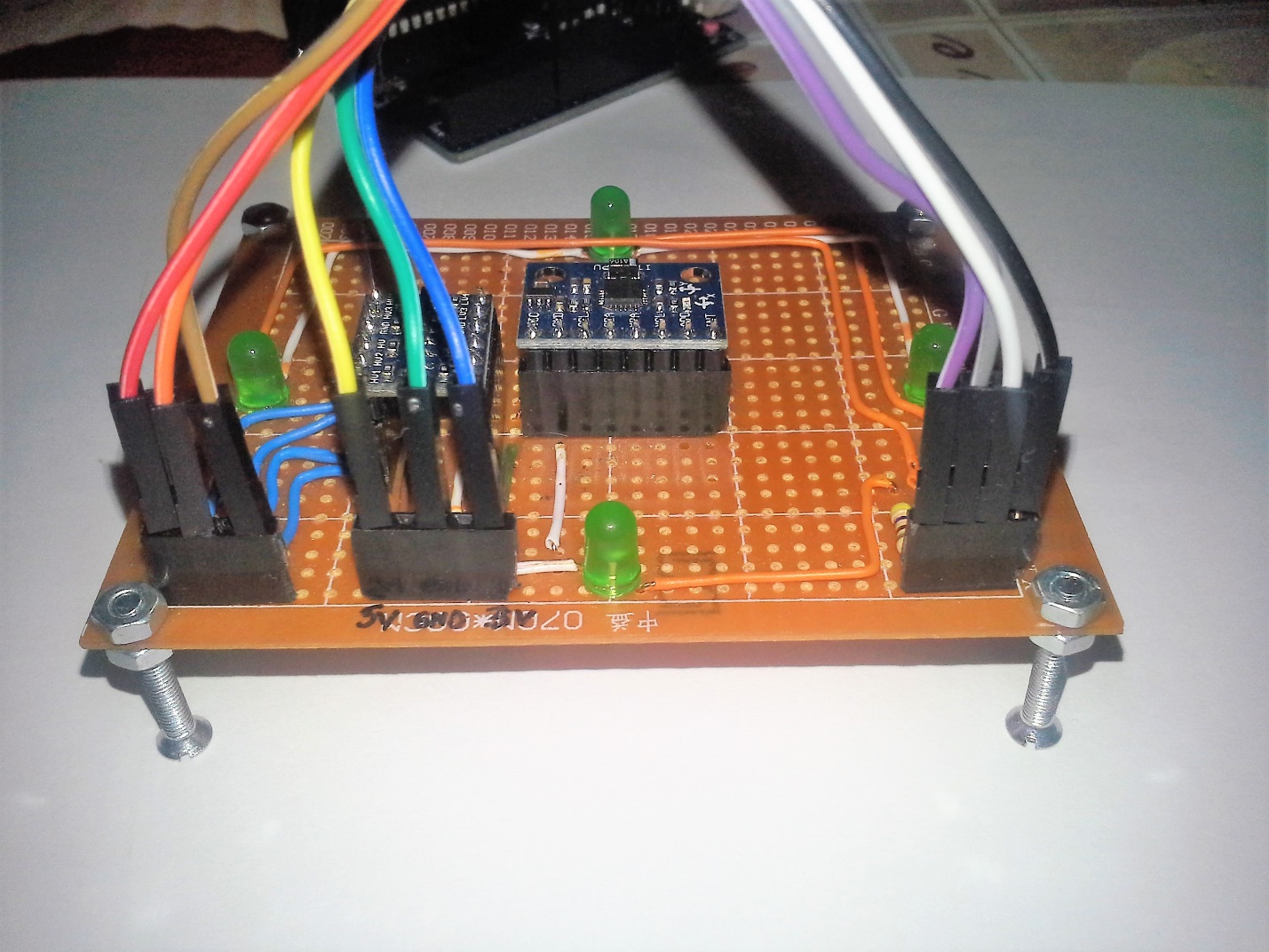
1. Conectati firele folosind imaginea de mai jos

RIGHT

BACK

FRONT

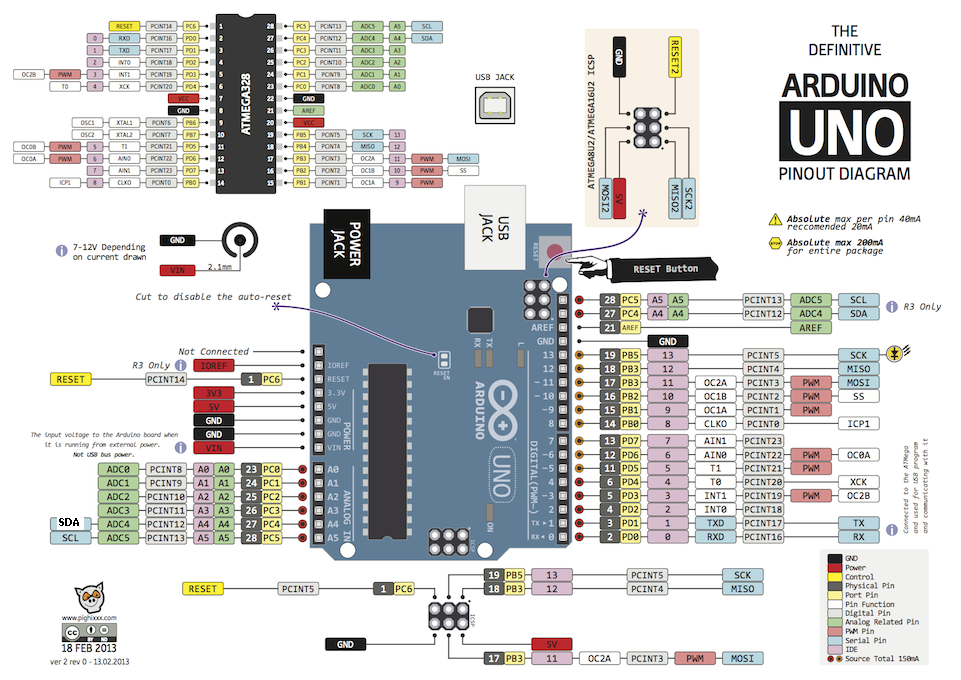
LEFT



**1 2 4 5 7 9 10 11 12 13**

|  |  |
| --- | --- |
| No.: | Color/Position |
| 1 | **SDA** |
| 2 | **SCL** |
| 3 | **-** |
| 4 | **INT** |
| 5 | **5V** |
| 6 | **-** |
| 7 | **GND** |
| 8 | **-** |
| 9 | **3V** |
| 10 | **BACK (PIN8)** |
| 11 | **LEFT (PIN9)** |
| 12 | **FRONT (PIN10)** |
| 13 | **RIGHT (PIN11)** |
|  |  |

1. Folosind tabelul de mai jos si imaginea de pinout Arduino conectati firele colorate si pe placa Arduino



1. Verificati conexiunile din nou si alimentati placa Arduino
2. Power LED-ul pe MPU6050 se aprinde
3. Folositi starter project (descrere in detaliu in capitolul urmator):
   1. Creati un proiect nou folosind AVR\_studio
   2. Introduceti in acest proiect fisierele din stater\_project
   3. Compilati si obtinexi .HEX
   4. Incarcati fisierul in microcontrollerul din Arduino
4. Dupa incarcarea fisierului .HEX se ruleaza o rutina de verificare (*MPU\_starter.mov*):
   1. Se aprind LED urile in succesiune: RIGHT -> FRONT -> LEFT -> BACK
   2. Deschideti un serial terminal (PUTTY): apar doua coloane de valori: prima coloana se modifica daca miscati placuta cu MPU6050, a doua coloana este 0
5. Daca nu constatati comportamentul de mai sus, verficati conexiunile( SDA/ SCL !)

# Pregatirea mediului software pentru lucrare de laborator

In aceasta lucrare veti folosi urmatoarele tooluri software:

* AVR Studio (ATMEL Studio) – IDE de dezvoltare software pentru microcontrollere
* PUTTY – serial terminal
* XLOADER – tool pentru incarcare fisier HEX in flash-ul microcontrollerului

Folositi fisierul **2\_software.pdf:**

* sa compilati starter codul(AVR Studio) care rezulta un HEX (punctual 1 din PDF)
* sa incarcati (XLoader) pe Arduino (punctual 3 din PDF).
* Comportamentul programului o sa verificati cu Putty (punctual 2 din PDF) si vizual.

# TEMA – 1

Initial axa y a accelerometrului este dezactivata,acesta masurand doar pentru axa x (in PUTYY avem o coloana de valori variabile si o coloana de zerouri).

Se cere sa activati axa y a accelerometrului.Pentru a realiza acest task trebuie modificata valoarea unui singur registru,registru care se gaseste in functia void MPU6050Init(): din fila **MPU6050.c.**

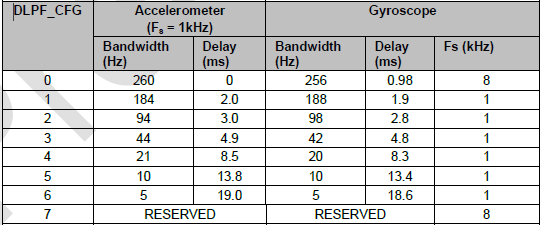
Folosindu-va de lista de mai jos rezolvati problema:

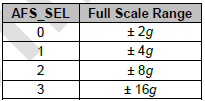
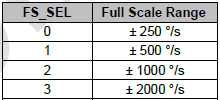
|  |  |  |  |
| --- | --- | --- | --- |
| ADDR | DENUMIRE REGISTRII | BIT | DESCRIERE |
| 6B | PWR\_MGMT\_1 | 7.DEVICE\_RESET | When set to 1, this bit resets all internal registers to their default values. |
| 6.SLEEP | When set to 1, this bit puts the MPU-60X0 into sleep mode. |
| 5.CYCLE | When this bit is set to 1 and SLEEP is disabled, the MPU-60X0 will cycle between |
| sleep mode and waking up to take a single sample of data from active |
| sensors at a rate determined by LP\_WAKE\_CTRL (register 108). |
| 3.TEMP\_DIS | When set to 1, this bit disables the temperature sensor. |
| [2:0].CLKSEL | 3-bit unsigned value. Specifies the clock source of the device. |
| 000-Internal 8MHz oscillator |
| 6C | PWR\_MGMT\_2 | [7:6].LP\_WAKE\_CTRL | 2-bit unsigned value. |
| Specifies the frequency of wake-ups during Accelerometer Only Low Power Mode |
| 5.STBY\_XA | When set to 1, this bit puts the X axis accelerometer into standby mode |
| 4.STBY\_YA | When set to 1, this bit puts the Y axis accelerometer into standby mode |
| 3.STBY\_ZA | When set to 1, this bit puts the Z axis accelerometer into standby mode |
| 2.STBY\_XG | When set to 1, this bit puts the X axis gyroscope into standby mode |
| 1.STBY\_YG | When set to 1, this bit puts the Y axis gyroscoper into standby mode |
| 0.STBY\_ZG | When set to 1, this bit puts the Z axis gyroscope into standby mode |
|  |
| 19 | SMPRT\_DIV | SMPLRT\_DIV[7:0] | The Sample Rate is determined by dividing the gyroscope output rate by this value |
| 1A | CONFIG | [5:3]EXT\_SYNC\_SET | Configures the FSYNC pin sampling. |
| [2:0]DLPF\_CFG | Configures the DLPF setting. |
| 1B | GYRO\_CONFIG | 7.XG\_ST | Setting this bit causes the X axis gyroscope to perform self test |
| 6.YG\_ST | Setting this bit causes the Y axis gyroscope to perform self test |
| 5.ZG\_ST | Setting this bit causes the Z axis gyroscope to perform self test |
| [4:3].FS\_SEL | 2-bit unsigned value 0- +-250 grade/s |
| 1C | ACCEL\_CONFIG | 7.XA\_ST | When set to 1, the X- Axis accelerometer performs self test |
| 6.YA\_ST | When set to 1, the Y- Axis accelerometer performs self test |
| 5.ZA\_ST | When set to 1, the Z- Axis accelerometer performs self test |
| [4:3]AFS\_SEL | Selects the full scale range of accelerometers.0-+-2g |
|  |  |
| 1.I2C\_MST\_RESET | This bit resets the I2 C Master when set to 1 while I2C\_MST\_EN equals 0. |
| This bit automatically clears to 0 after the reset has been triggered. |
| 0.SIG\_COND\_RESET | When set to 1, this bit resets the signal paths for all sensors (gyroscopes, |
| accelerometers, and temperature sensor). This operation wil also clear the |
| sensor registers. This bit automatically clears to 0 after the reset has been triggered. |

|  |  |
| --- | --- |
| **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) |
|  |  |

|  |  |
| --- | --- |
| **CLKSEL** | **Clock Source** |
| 0 | Internal 8Mhz oscillator |
| 1 | PLL with X axis gyroscope reference |
| 2 | PLL with Y axis gyroscope reference |
| 3 | PLL with Z axis gyroscope reference |
| 4 | PLL with external 32.768kHz |
| 5 | PLL with external 19.2 Mhz reference |
| 6 | Reserved |
| 7 | Stops the dock and keeps the timing generator in reset |

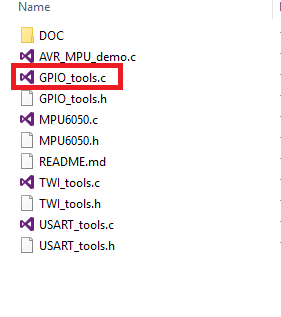
|  |
| --- |
| Sample Rate = Gyroscope Output Rate/(1+SMPLRT\_DIV) where |
|
| Gyroscope Output Rate= 8kHz when the DLPF is disabled(DLPF\_CFG=0 or 7),and 1kHz |
| when the DLPF is enabled |





# TEMA – 2

In fisierul **GPIO\_tool.c** se gasesc 3 subpuncte ale acestei teme si cerintele impuse.



Ca obiectiv in tema 2, o sa avem de facut cateva modificari deoarece dupa rezolvarea temei 1 pe USART apar valori pentru X si Y. Exista in fisierul specificat functia **driveLED** pentru prelucrarea valorilor pe axa X (Xaxis).

*Cerinte:*

2.1

* Modificati codul sa functioneze si pentru axa Y (yAxis)

In functia driveLED este deja implementat algorimul pentru axa X. Studiati acest algorimt si implementati pentru axa Y. Atentie xAxis si yAxis sunt pointeri!

2.2

* Explicati de ce sunt folosite in codul nostru (MIN\_POS\_VAL, MAX\_POS\_VAL, MIN\_NEG\_VAL, MAX\_NEG\_VAL)

Valorile citite sunt pe 16 biti cu semn. Valorile negative sunt exprimate in C2. Care este valoarea maxima pozitiva atunci? Cum este reprezentata binar aceasta valoare?

Care este valoarea cea mai mica negativa ? cum este reprezentata acea valoare in binar?

Raspunsurile le gasiti in GPIO\_tools.h

OBS.: sa folosit un -0x800 si +0x800 in definitia valorilor MIN\_POS\_VAL ….

Cum scrie si in cod acea modificare s fost pentru un raspuns mai neted a placutei cu LED uir la valorile accelerometrului

2.3

* De ce sunt folositi pointeri ca parametru?

In primul rand pointerul este o variabila care contine o adresa de memorie. Aceasta variabila contine adresa unei variabile.

Avantajele pointerilor sunt:

- ofera posibilitatea de a modifica argumentele de apelare a functiilor;

- permit o alocare dinamica a memorie;

- pot imbunatatii eficienta anumitor rutine.

In codul nostru s-a folosit pointer in parametru pentru a economisii memorie, pentru a nu se face copie la un anumit parametru.