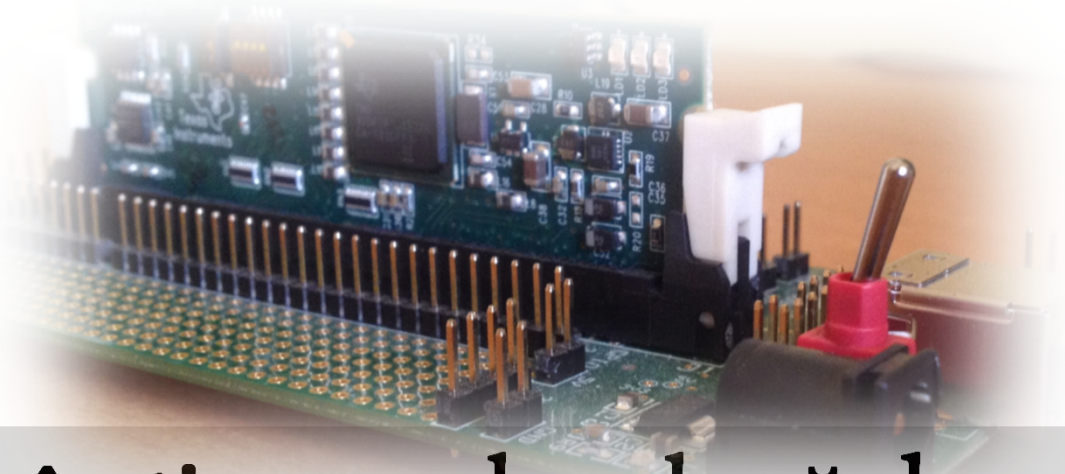
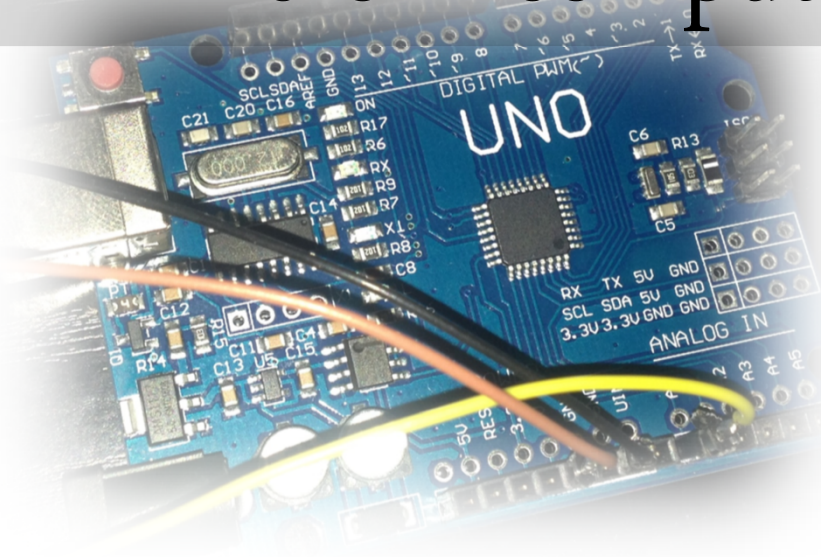


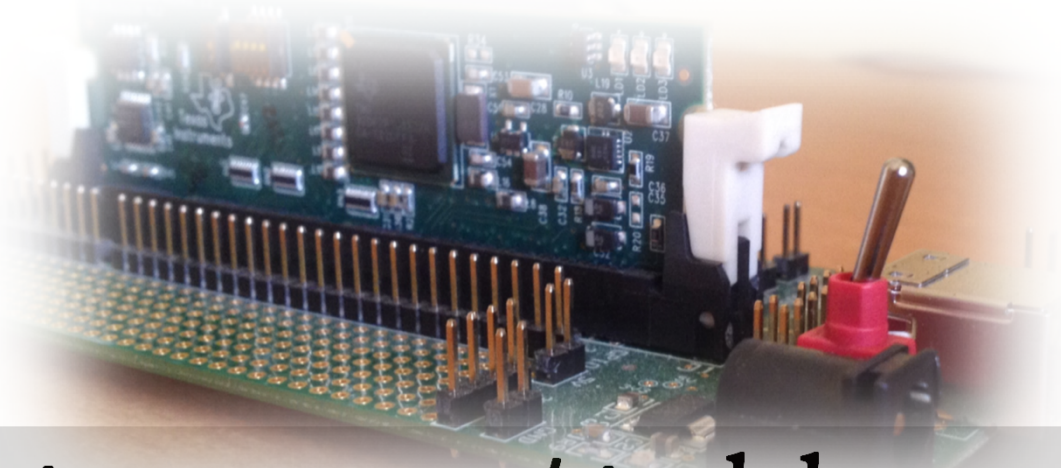
Sisteme de calcul în timp real



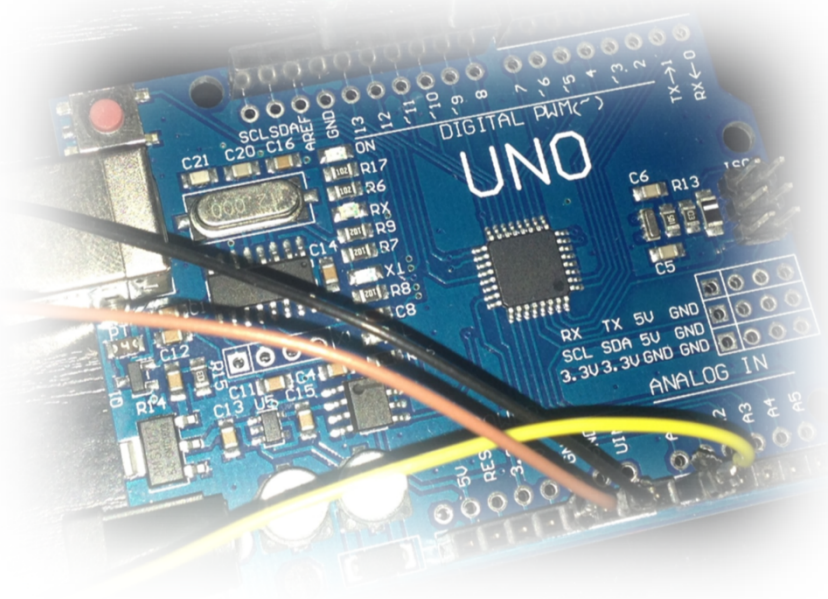
Sisteme de calcul în timp real pe bază de
micro – computer și sistem de operare



Sisteme de calcul în timp real



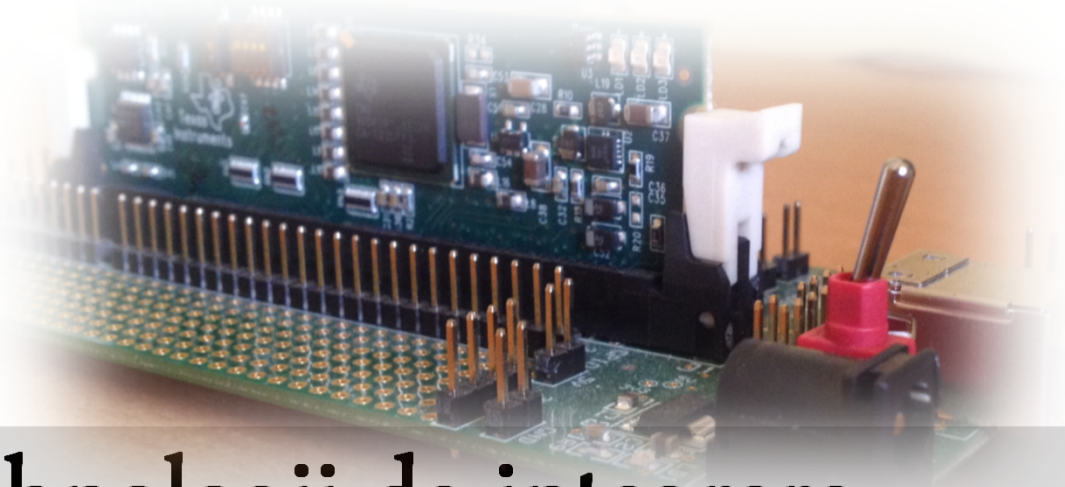
Sisteme de calcul încorporate / înglobate



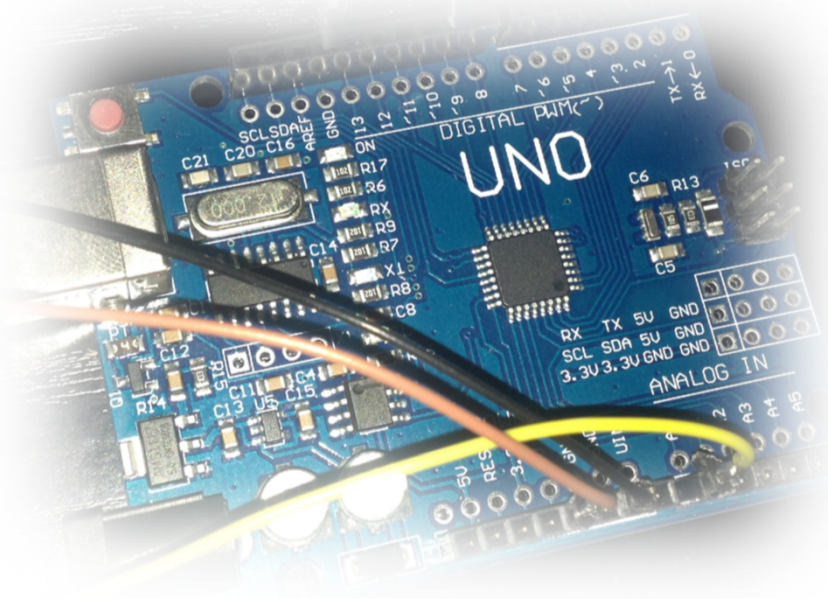
Micro-sisteme încorporate sau înglobate

- Conceptul de „încorporare” sau „înglobare” (eng. embedded system) reflectă faptul că un sistem de calcul în timp real nu este altceva decât o soluție tehnică la problema integrării unui sistem distribuit de circuite digital cu funcții dedicate specifice (ex. convertor analog – digital, interfață de comunicare serial, unitate aritmetico – logică etc...)
- În acest sens se recurge la adoptarea unei soluții în procesul de fabricare a unui sistem de calcul și control. Deci, în mod automat apar unele beneficii dar și neajunsuri precum:
 - ✓ Gradul mare de integrare (reducerea costurilor, materialului);
 - ✓ Diversitatea funcțională a unor componente (multifuncțional);
 - ✓ Reconfigurabilitatea (mai mult logic - soft, mai puțin fizic - hard);
 - ✓ Performanța distribuită (contrar unui dedicat – ex. placa video);
 - ✓ Gestionarea nivelului de putere absorbit / debitat;
 - ✓ Inter – dependență funcțională (incapacitatea de a utiliza unele circuite cât timp sistemul de calcul realizează o anumită funcție);

Sisteme de calcul în timp real



Exemple de tehnologii de integrare



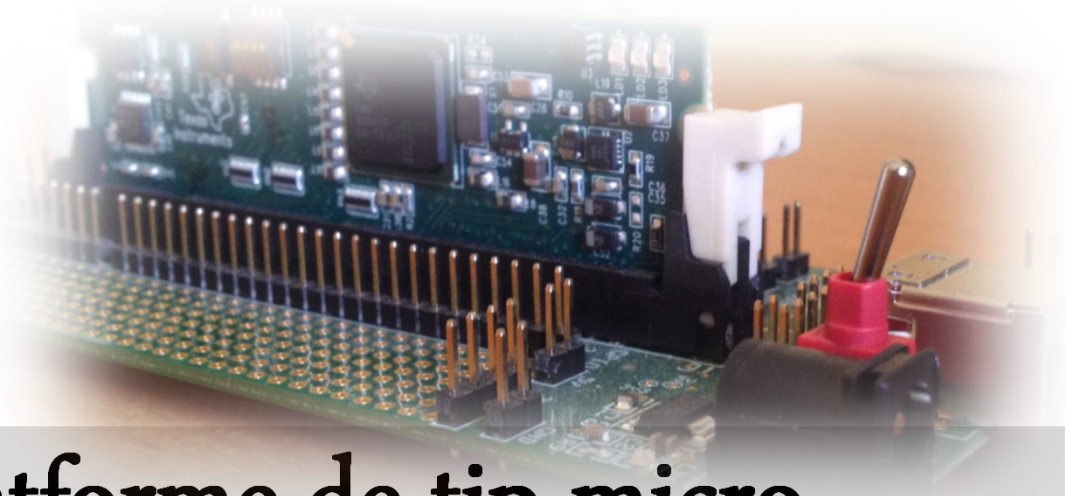
Tehnologia ON-BOARD

- Tehnologia ON-BOARD – Se referă la integrarea anumitor module de interfațare (cu funcția de intrare / ieșire) în structura generală a sistemului de calcul (ex. modulul grafic sau modulul audio integrat în placa de bază);
- AVANTAJE:
 - ✓ Reducerea costurilor de implementare;
 - ✓ Simplificarea operațiunilor de gestionare a resurselor (la comun);
 - ✓ Inter-operabilitatea (ex. realizarea calculelor în placa video);
- DEZAVANTAJE:
 - ❖ Performanțele depind de întreg sistemul;
 - ❖ Modulele integrate nu au resurse proprii (memorie RAM video);
 - ❖ Modulele ON-BOARD NU POT deservi DOAR funcția pentru care au fost concepute;

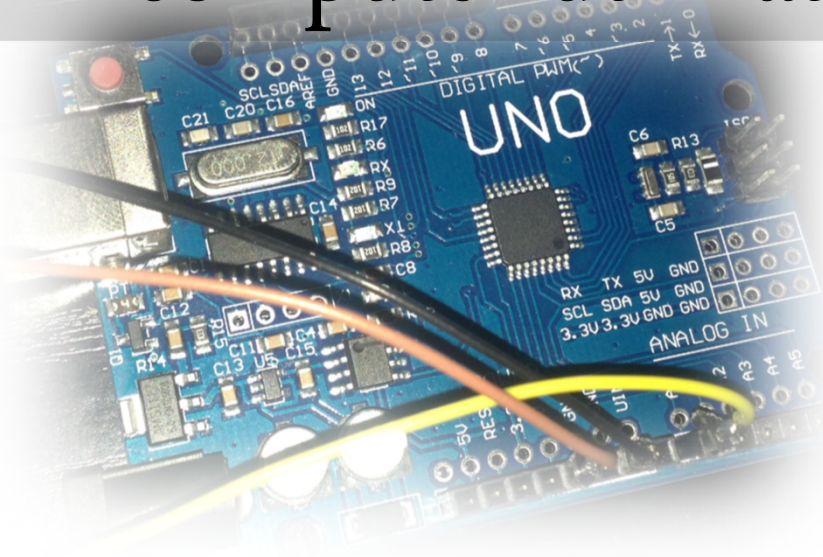
Tehnologia System-on-Chip

- Tehnologia System-on-Chip (SoC) – Se referă la integrarea mai multor structuri mai complexe (precum un microprocesor și o memorie), într-un singur cip (sub aceeași pastilă). Microprocesorul în sine, este un sistem complex format din circuite logice simple, la fel și memoria sau o arie de porți programabilă (ex. Field Programmable Gate Array);
- AVANTAJE:
 - ✓ Reducerea costurilor de implementare;
 - ✓ Simplificarea logicii de control (ex. software – alocarea adreselor);
 - ✓ Domeniul larg de aplicabilitate al unui singur cip;
- DEZAVANTAJE:
 - ❖ Impedimentele termice (ex. încălzirea pastilei);
 - ❖ Nu permite flexibilitatea gestiunii separate pe bază de soft (ex. dacă se dorește creșterea unor anumite performanțe, trebuie să se renunțe la anumite funcții ex. frecvența PWM și interfața I2C);
 - ❖ Defectarea unui singur modul, afetează întreg cipul;

Sisteme de calcul în timp real



Exemple de platforme de tip micro –
computer utilizate în Ingineria Electrică



Raspberry Pi 3 Model B



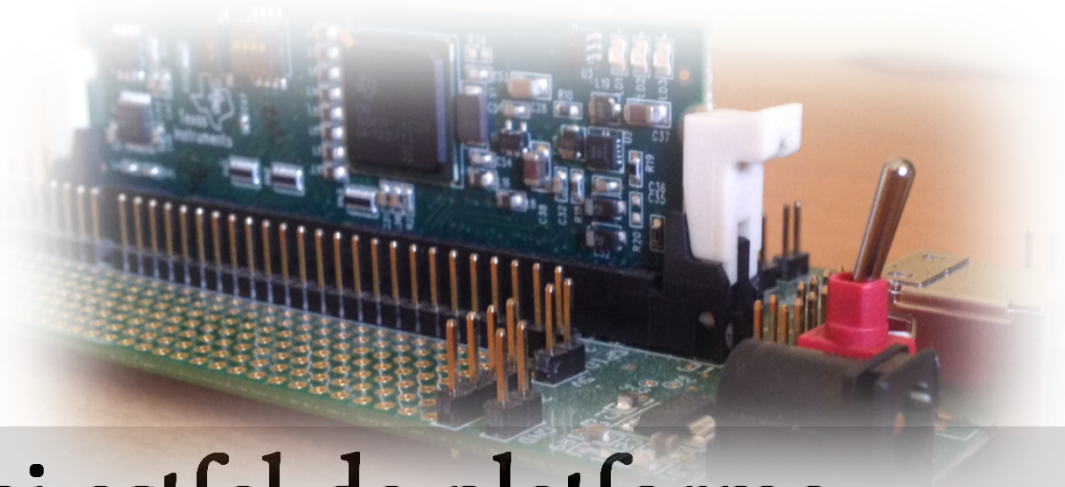
<https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>

Intel Galileo Generația I

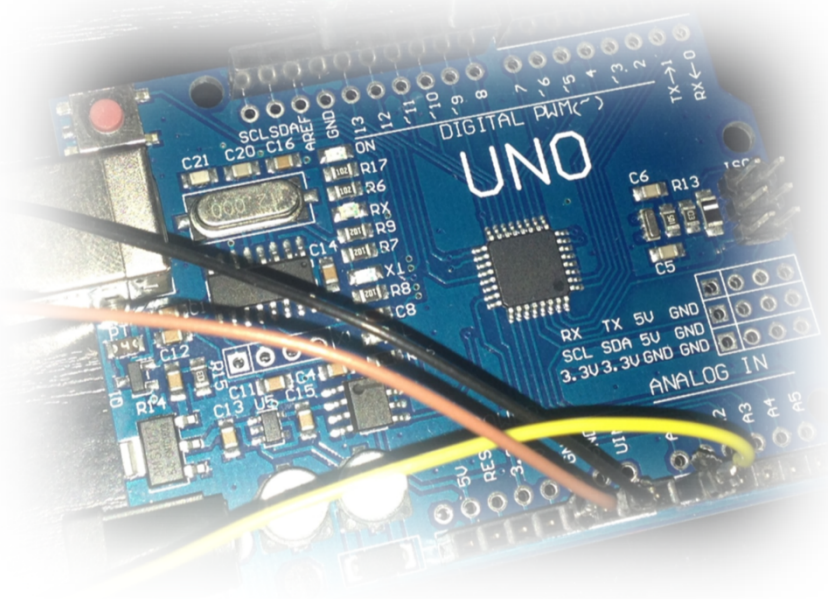


<https://www.generationrobots.com/en/401641-arduino-intel-galileo-board.html>

Sisteme de calcul în timp real



Utilizarea unei astfel de platforme



Platforma Intel Galileo

- Deservește scopul unui micro – computer industrial pentru interfațarea proceselor fizice, și gestiunea de la distanță;
- Acest lucru îl poate realiza prin intermediul modulelor de interfațare (Ethernet, GPIO, ADC, PCI, USB etc...) și a sistemului de operare instalat pe cardul de memorie;
- Sistemele de operare care pot rula pe această platformă sunt: Linux (Yocto / Clanton, openSUSE, Debian), Microsoft Windows 8.1 (embedded - core edition);
- Gestiunea resurselor se realizează prin intermediul protocoalelor:
 - ✓ SSH (Secure Shell);
 - ✓ FTP (File Transfer Protocol);
 - ✓ Remote Desktop / VNC;
 - ✓ HTTP / WEB;

Platforma Intel Galileo

- Avantajul major pe care îl prezintă un astfel de sistem constă în faptul că, acesta, poate fi utilizat:
 - Atât ca și un dispozitiv dedicat unei singure aplicații (ex. micro-controller cu funcții mai avansate);
 - Cât și ca un dispozitiv centralizator de informații (ex. calculatorul de gestiune a procesului, server de date etc...);
- Astfel, sistemul Intel Galileo, nu este limitat la un mod de lucru anume sau la un limbaj de programare anume;
- Totuși, există cinci direcții de abordare a procesului de dezvoltare a aplicației pe o astfel de platformă:
 1. Limbajul Wiring (Arduino IDE);
 2. Dezvoltarea directă a aplicației (ex. Python, Java, C++ etc...);
 3. Eclipse Studio;
 4. Intel System Studio (Docker);
 5. Visual Studio .NET – Windows 8.1;

Platforma Intel Galileo

- Limbajul Python spre exemplu, este utilizat des, în dezvoltarea aplicațiilor pentru sistemele de calcul în timp real (ex. dSPACE, Typhoon HIL, PLECS RTBox etc.);
- În acest sens, vom aborda această variantă de lucru;
- Aspectele necesare sunt:
 - ✓ Instalarea aplicațiilor Putty și WinSCP pe calculatorul personal;
 - ✓ Instalarea sistemului Intel IOT Development Kit pe Intel Galileo;
- ✓ Dezvoltarea aplicațiilor pe baza bibliotecii MRAA:
- ✓ <http://iotdk.intel.com/docs/master/mraa/python/index.html>

Sisteme de calcul în timp real

Conținutul uzual pentru un fișier Python:

#!/usr/bin/python - pentru a specifica o cale absolută (eng. absolute path) în Unix, anume înspre directorul ce conține bibliotecile interpretorului de cod - Python;

#<ceva_de_spus> - introducerea unor indicații în interiorul codului de program;
Frazele care încep cu semnul diez „#” nu sunt luate în considerare!

<instrucțiune_specifică_limbajului> <argument> - exemplu: **print** "Text pentru afișare în consolă!"

OBSERVAȚII:

1. Instrucțiunea „**print**” este specifică limbajului de programare;
2. „Text pentru afișare în consolă!” este argumentul instrucțiunii;
3. Delimitările ierarhice (eng. code blocks) se realizează prin **aliniere la stânga** cu **trei spații** (indent)!

Semnalizarea intermitentă

<code>#!/usr/bin/python</code>	#Specificare cale absolută interpretor Python;
<code>import mraa</code>	#Importare bibliotecă "mraa";
<code>import time</code>	#Importare bibliotecă "time";
<code># Setup</code>	#Comentariu;
<code>x = mraa.Gpio(9)</code>	#Adresarea ieşirii digitale;
<code>x.dir(mraa.DIR_OUT)</code>	#Configurarea modului de lucru al ieşirii;
<code># Loop</code>	
<code>while True:</code>	#Bucă „WHILE”;
<code>x.write(1)</code>	#Activarea ieşirii;
<code>time.sleep(0.5)</code>	#Aşteptare 500 milisecunde;
<code>x.write(0)</code>	#De-activarea ieşirii;
<code>time.sleep(0.5)</code>	#Aşteptare 500 milisecunde;

Preluarea semnalelor analogice

```
#!/usr/bin/python    #Specificare cale absolută interpretor Python;  
import mraa         #Importare bibliotecă "mraa";  
import time         #Importare bibliotecă "time";  
  
try:                #Directivă de pre-procesare;  
    x = mraa.Aio(0)  #Inițializare intrare analogică;  
    print(x.read())  #Afișarea valorii citite ca și număr întreg;  
    print("%.5f" % x.readFloat()) #Afișare ca număr fracționar;  
except:              #Directivă de pre-procesare;  
    print("Is this correct?") #Afișare mesaj aleatoriu;
```


Generarea semnalelor modulate în lăţime

<code>#!/usr/bin/python</code>	#Specificare cale absolută interpretor Python;
<code>import mraa</code>	#Importare bibliotecă "mraa";
<code>import time</code>	#Importare bibliotecă "time";
<code>x = mraa.Pwm(3)</code>	#Comutare ieşire digitală în mod PWM;
<code>x.period_us(700)</code>	#Configurarea perioadei de comutaţie;
<code>x.enable(True)</code>	#Activarea ieşirii;
<code>value = 0.0</code>	#Inițializare valorii factorului de umplere;
<code>while True:</code>	#Bucla „WHILE”;
<code>x.write(value)</code>	#Stabilirea factorului de umplere;
<code>time.sleep(0.05)</code>	#Întârziere;
<code>value = value + 0.01</code>	#Incrementare factor de umplere;
<code>if value >= 1:</code>	#Re-inițializare la depășire de scală;
<code>value = 0.0</code>	