

Structuri de date: clustere

Utilizarea cluster-elor

LabVIEW permite realizarea structurilor de date prin utilizarea cluster-elor.

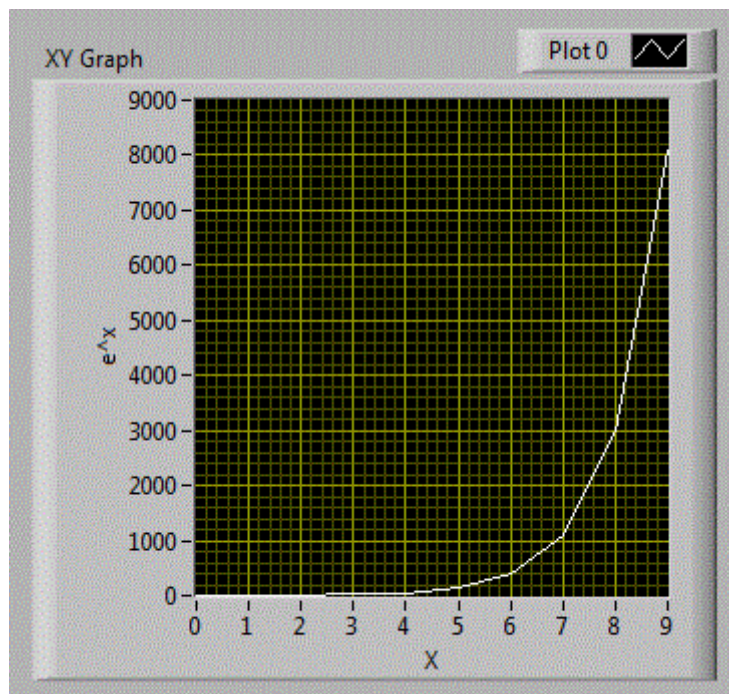
Controalele grafice de tip "Graph" servesc pentru reprezentarea grafică a diverselor tipuri de date. De multe ori reprezentarea datelor este complexă și nu presupune numai reprezentarea evoluției în timp a unei mărimi. Pentru a utiliza majoritatea controalelor de tip "Graph", datele trebuie pregătite în structuri specifice fiecărui control "Graf". Este nevoie deci de a realiza diverse structuri de date în vederea utilizării controalelor "Graph". Structurile de date sunt realizate utilizând clustere. Clusterelor sunt funcții grupate în :

Function-->Programming-->Cluster, Class & Variant

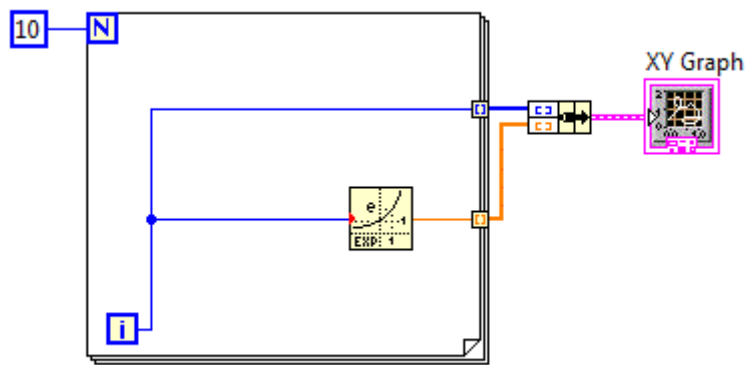
Realizarea unei structuri

- **Afisarea în coordonate x, y.**

Vom realiza pentru o aplicație pentru a fișarea funcției $y=e^x$ în coordonate x,y [cluster v01](#) în care să se realizeze o structură de date necesară controlului grafic "XY Graph" aflat în grupul Controls-->Modern-->Graph-->XY Graph .

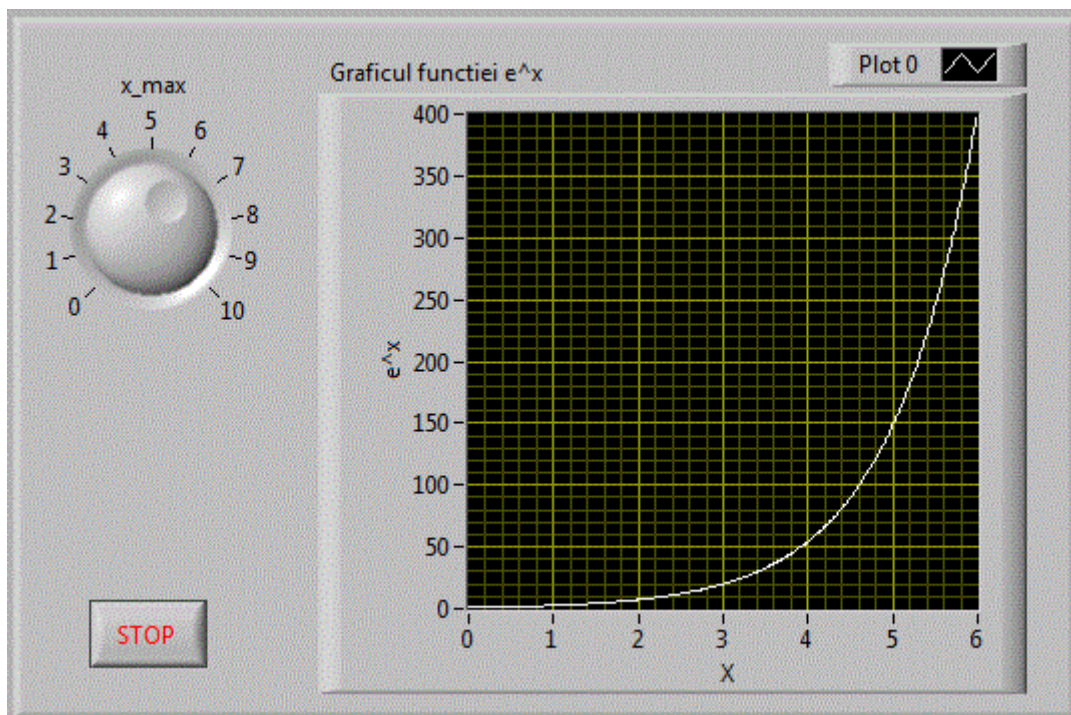


Vom crea o structură de date în care să fie cuprins un vector cu elementele lui x și un vector cu elementele lui y adică $y=e^x$.

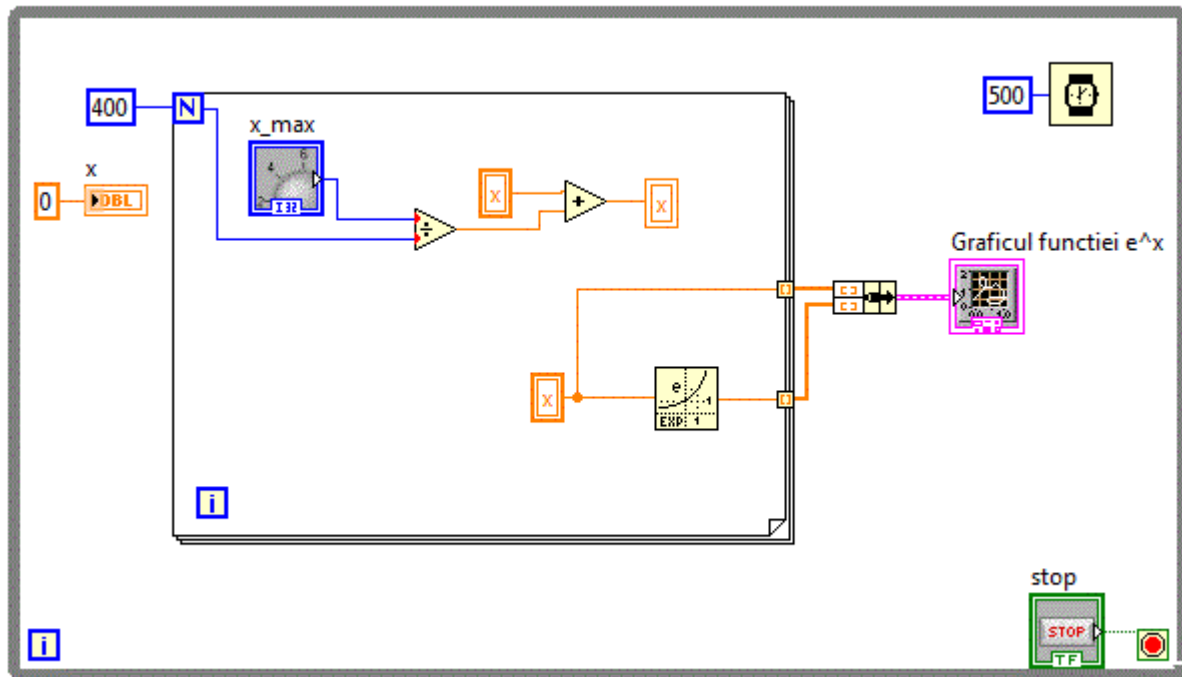


Structura de date necesara conrolului grafic XY Graph este realizata utilizand un cluster folosind "Bundel Function" pe care o gasim in Function-->Programming-->Cluster, Class & Variant-->Bundle.

Dupa cum se observa, calitatea graficului nu este prea buna pentru ca am afisat numai 10 puncte. Pentru o calitate mai buna, avem nevoie de cel putin 100 de puncte. Daca vrem sa pastram domeniul de definitie pentru x, intre 0-10, va trebui sa introducem o variabila noua numita x, deoarece nu mai putem folosi variabila i pe post de x. Obtinem astfel aplicatia : [cluster_v02](#) in care se afiseaza 100 de puncte, fara a modifica domeniul de definitie 0-10.



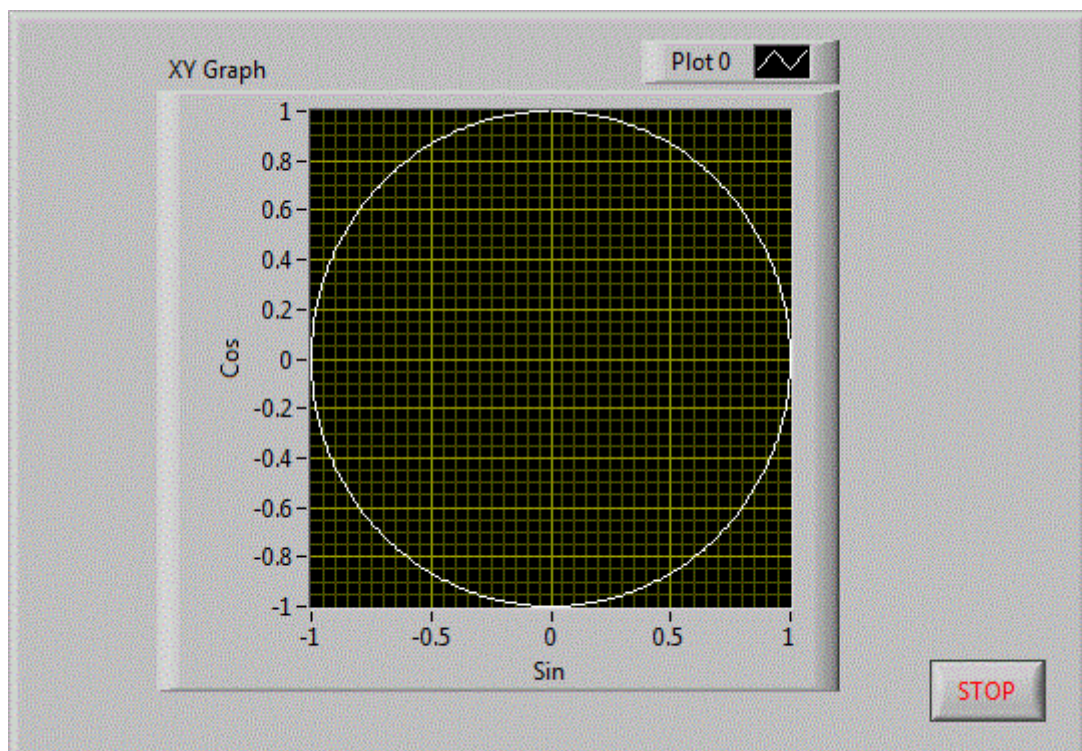
In diagrama bloc se va introduce variabila locala x.



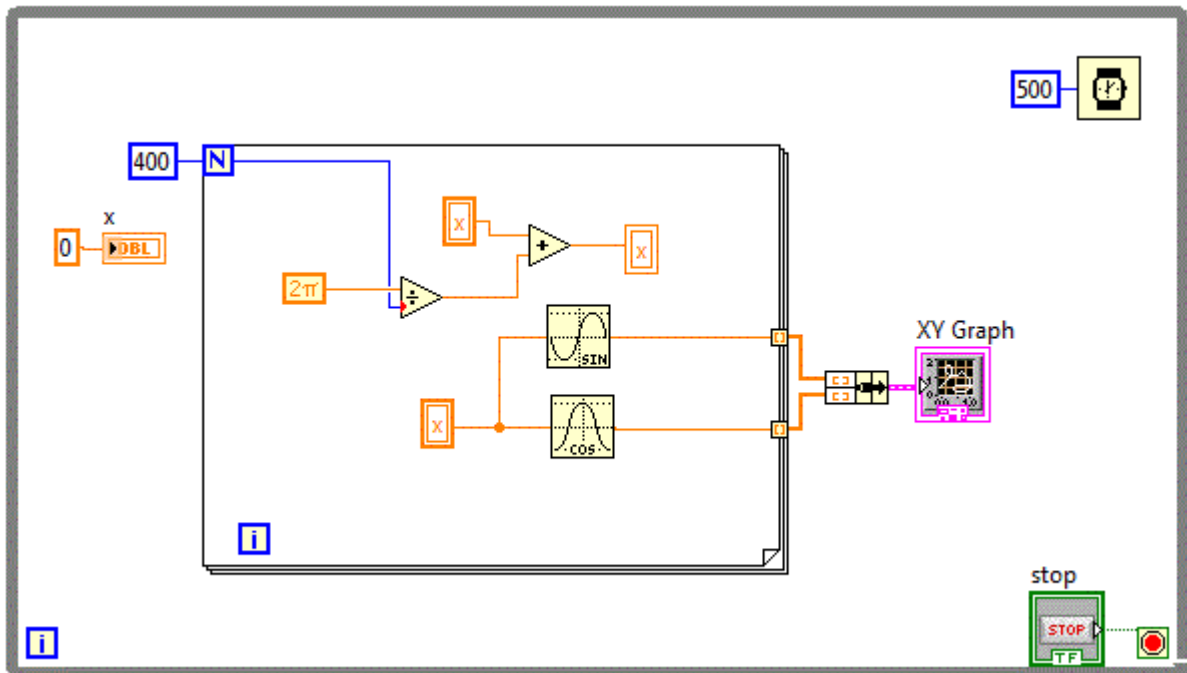
Pentru functia anterioara $y=e^x$, afisarea graficului functiei se putea realiza si prin intermediul unui control grafic "Waveform Graph", nemaifiind necesara definirea unei structuri de date.

Exista functii care sunt mult mai usor de reprezentat in coordonate polare. Astfel un cerc este mult mai simplu de reprezentat. Chiar si in coordonate carteziene XY este mai usor de reprezentat folosind ecuatiile: $x=\sin(t)$; $y=\cos(t)$ insa pentru aceasta avem nevoie de un control XY Graph. Aceasta metoda de reprezentare grafica a unui cerc este utilizata in aplicatia:

[cluster_v03](#)



Metoda utilizata pentru reprezentarea functiei e^x s-a folosit si pentru reprezentarea grafica a unui cerc.



- **Figurile lui Lissajous**

Figurile lui Lissajous sunt grafice care rezulta din compunerea a doua functii sinus sau cosinus de diferite frecvente. Numarul de curbe inchise depinde de raportul dintre frecventele celor doua functii sinus. [cluster v04](#)

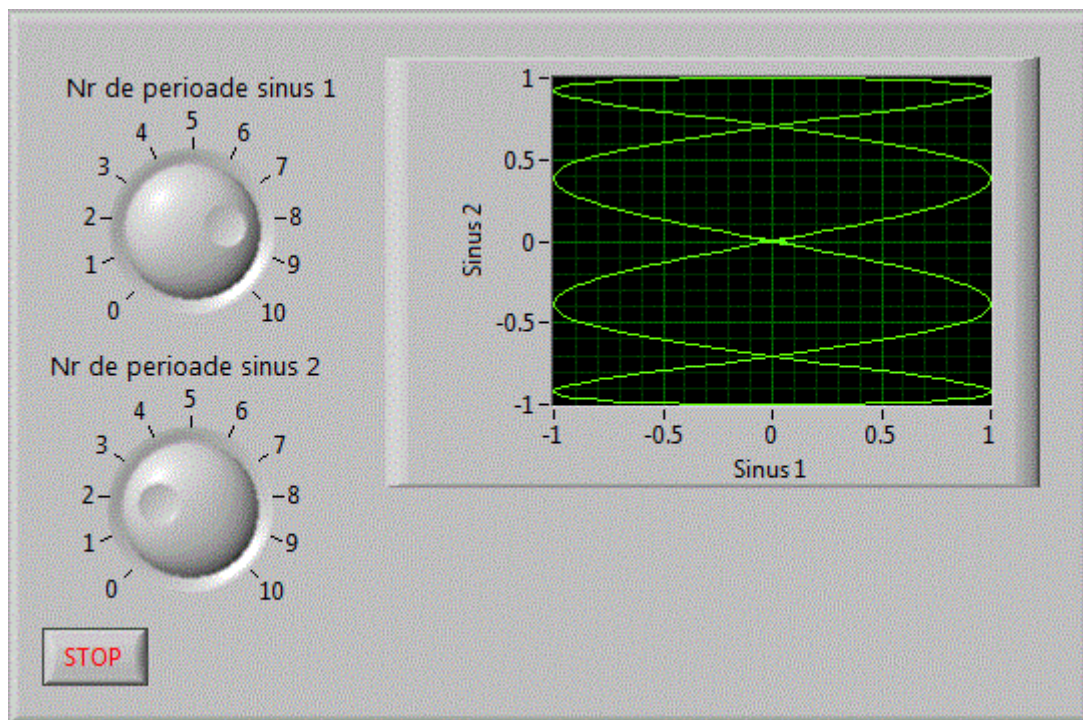
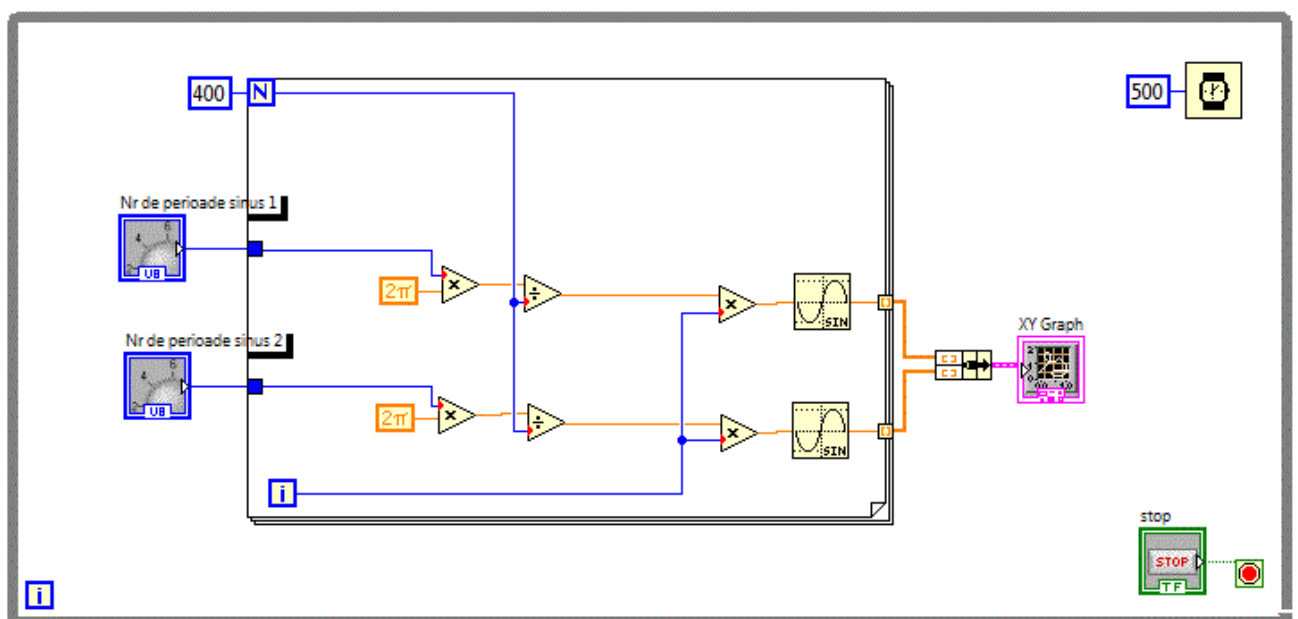


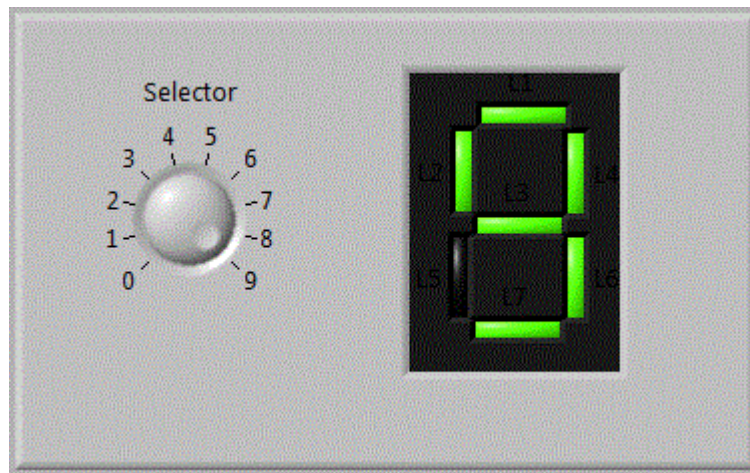
Diagrama bloc fiind asemanatoare cu diagrama anterioara, cu deosebirea ca se pot ajusta frecventele celor doua functii sinus.



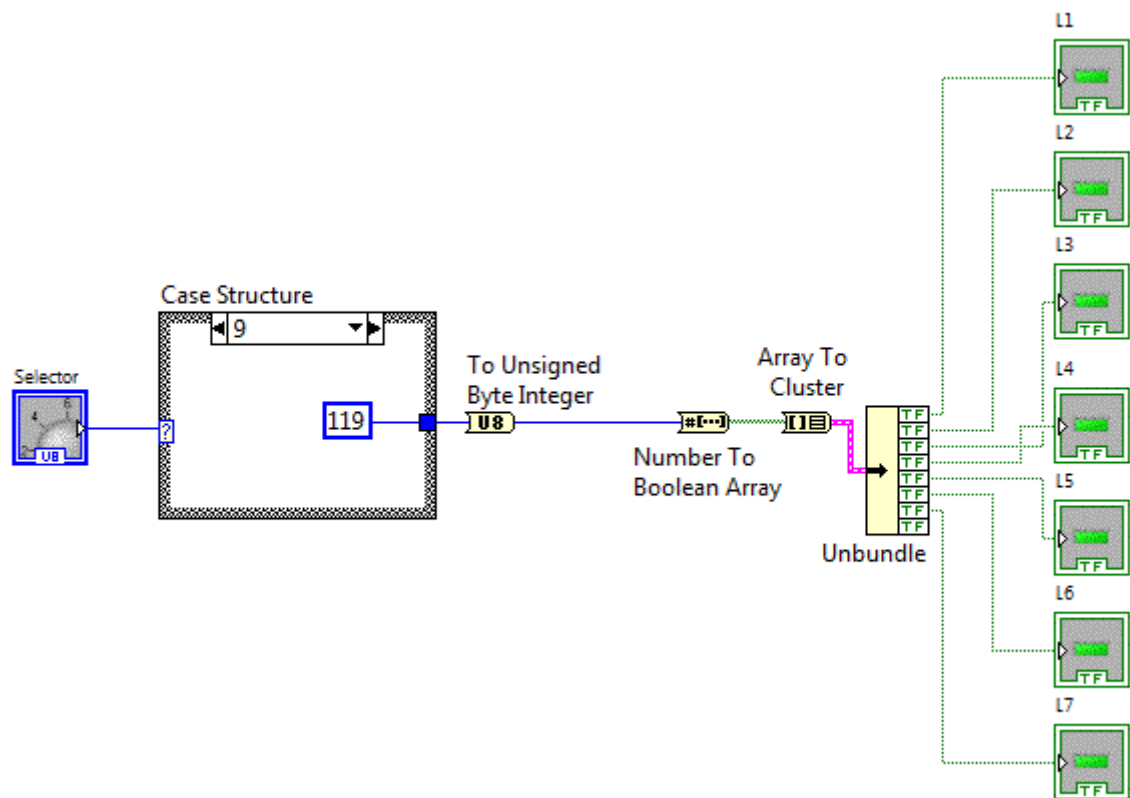
Descompunerea unei structuri in componente

- **Afisarea binara.**

Vom relua pentru inceput, o aplicatie pentru conversie binara si fisare pe 7 segmente
[cluster_v00](#)



In diagrama bloc se va utiliza o structura de date creata cu functia "Unbundle Function" folosind "Bundel Function" pe care o gasim in Function-->Programming-->Cluster, Class & Variant-->Unbundle.



Structura "Case" ne ofera un numar int a carui reprezentare binara o reprezinta combinatia de segmente care trebuie activate pentru a forma afisarea corespunzatoare pe 7 segmente. Numarul int se converteste in Uint dupa care e convertit intr-un vector binar. In urma conversiei "Array to Cluster" obtinem o structura de date pe care daca o despachetam cu functia "Unbunde" obtinem in sfarsit 7 iesiri binare necesare comandarii celor 7 segmente.

- **Roza polara**

Functia:

- $r(t)=a*\cos(3*t+fi)$

se mai numeste roza polara cu 3 petale.

Roza polara este o curba matematica celebra care arata ca o floare cu petale si care poate fi exprimata ca o ecuatie polara simpla, de forma $r(t)=a*\cos(n*t)$.Daca n este intreg, aceasta ecuatie produce o roza cu n petale, daca n este impar, sau cu 2n petale daca este par. Daca n este rational dar nu intreg, o forma asemanatoare cu roza ar putea aparea, dar va avea petale suprapuse.

Tinand cont de ecuatiile de transformare in coordonate carteziene:

- $x=r*\cos(t)$
- $y=r*\sin(t)$

, vom obtine expresiile lui x si y de forma:

- $x=\cos(3*t)*\cos(t);$
- $y=\cos(3*t)*\sin(t);$

unde t ia valori intre 0 si 2pi adica 0 si 6.3 radiani Aplicatia: [cluster_v05](#) incearca sa traseze grafic roza polara.

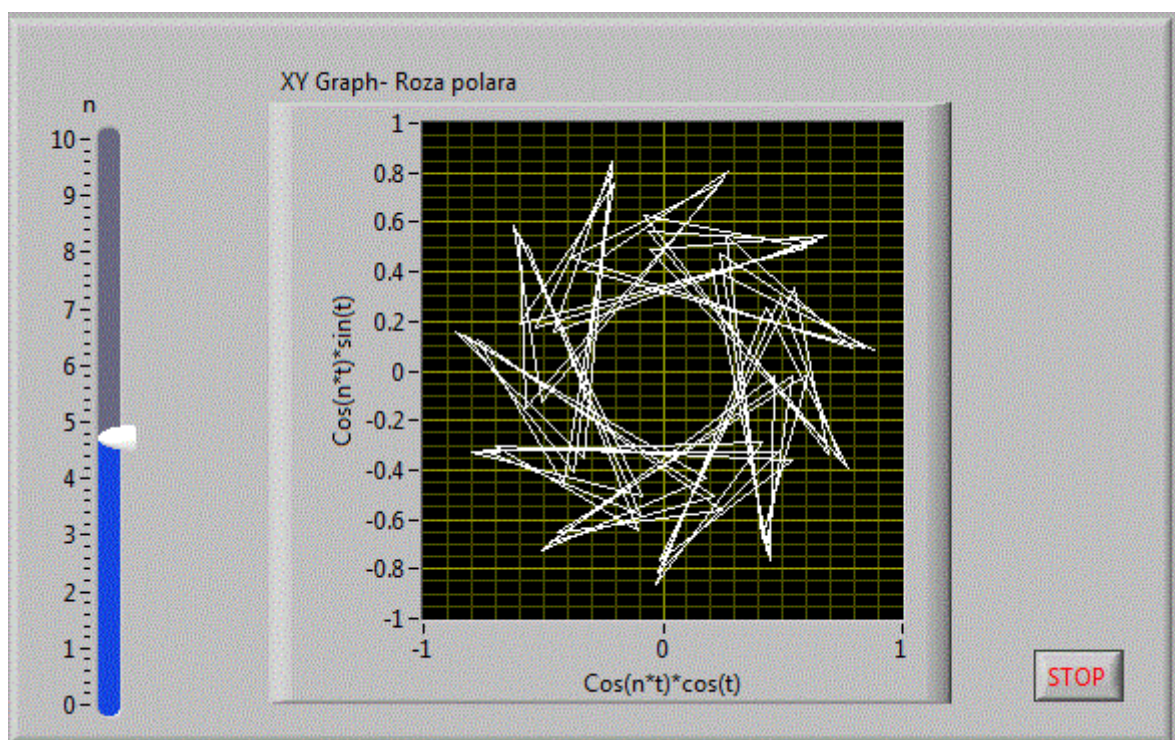
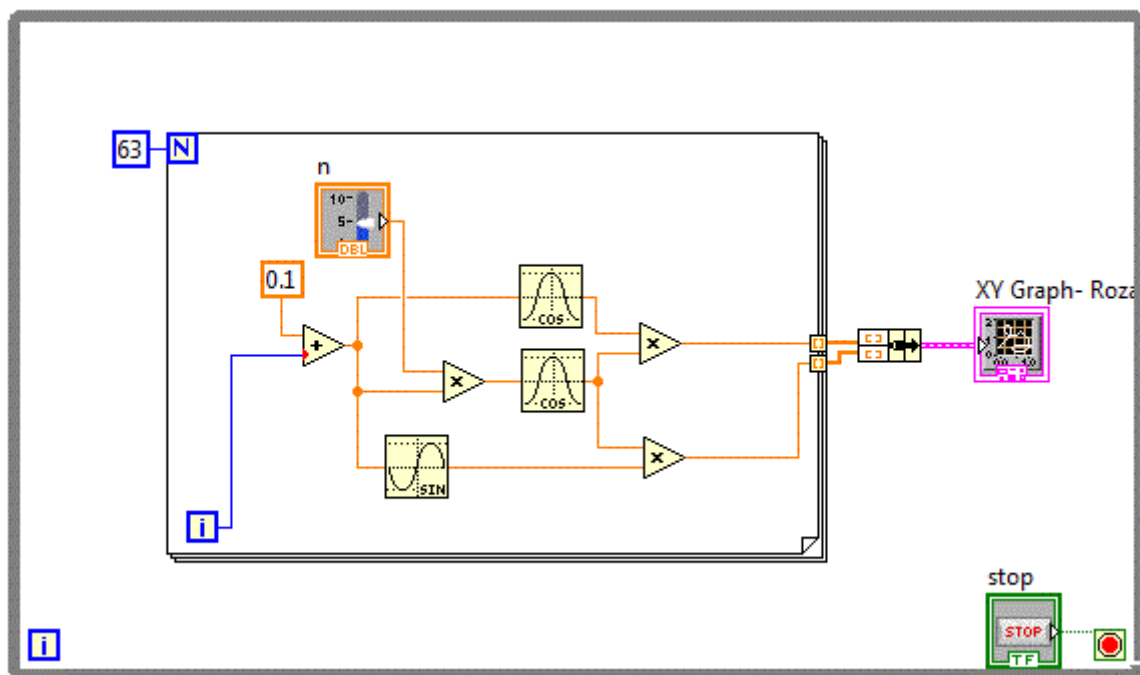


Diagrama bloc fiind:



Imaginea de sus nu prea seamana a roza polara, desi ecuatiile functiei sunt corecte. Din cauza "pasului" destul de grosier al unghiului alfa, reprezentarea nu este corecta.

In urmatoarea aplicatie: [cluster_v07](#) se foloseste o varabila separata pentru alfa reusindu-se sa se traseze grafic roza polara.

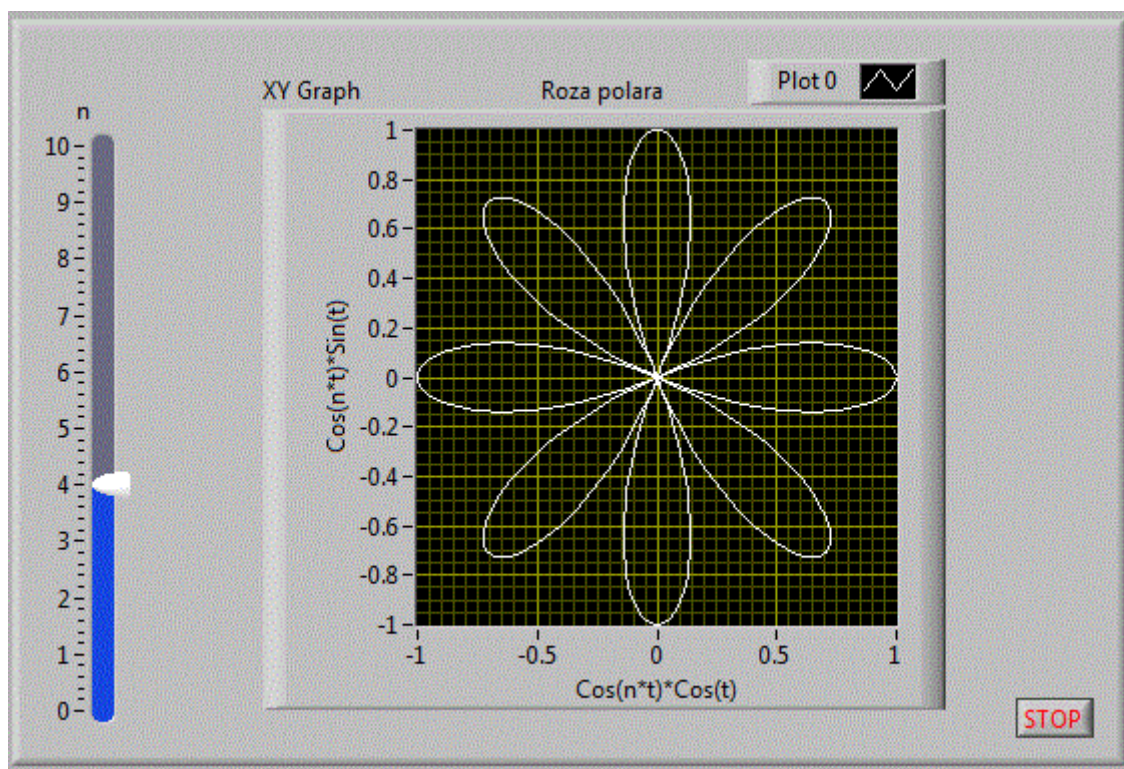
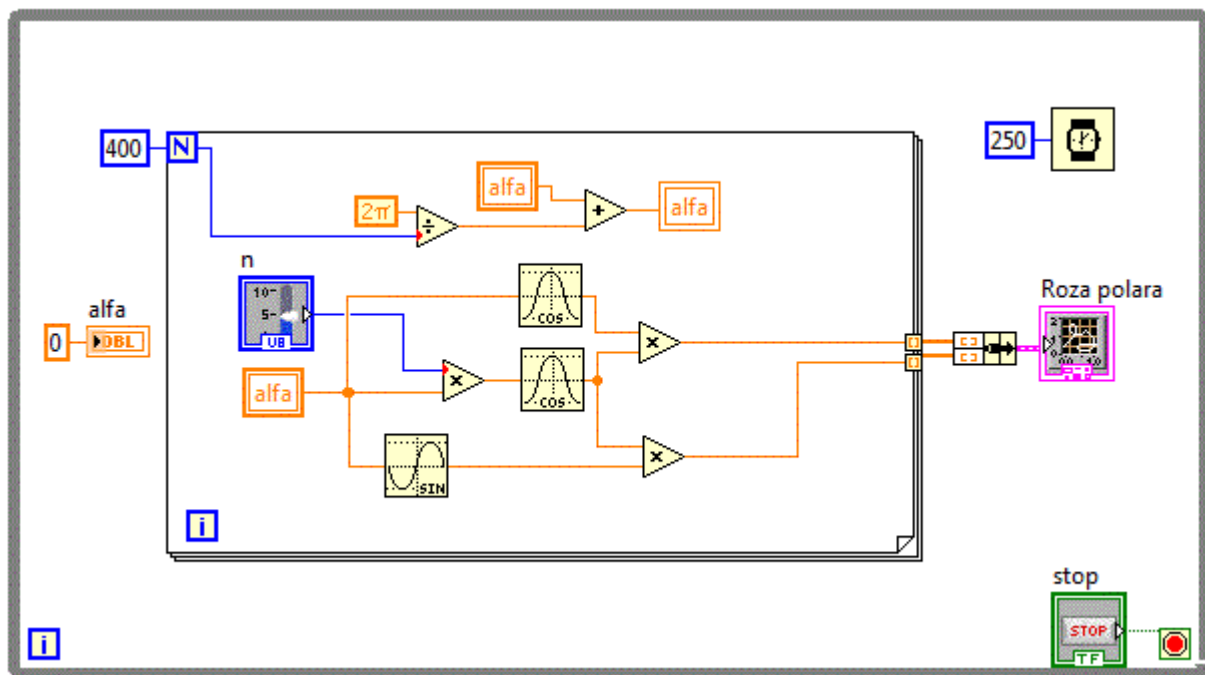


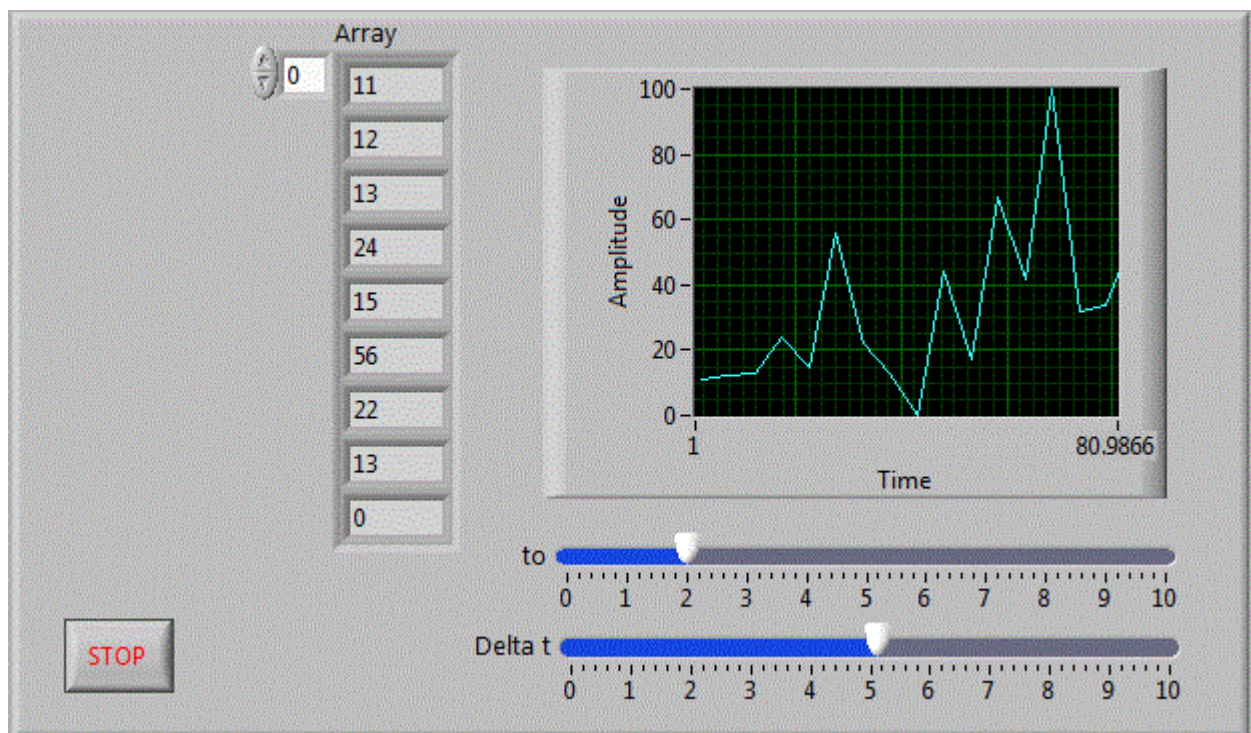
Diagrama bloc fiind:



• Utilizare avansata Waveform Graph

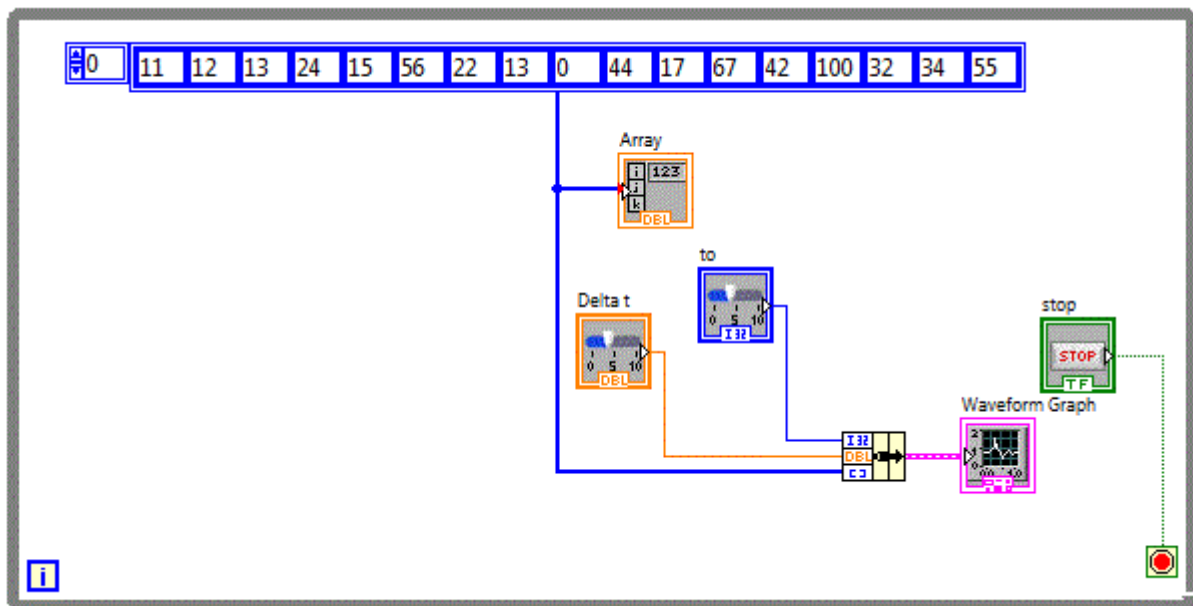
Am utilizat pana acum controlul grafic Waveform Graph insa nu am exploatat la maxim facilitatile acestuia.

In urmatoarea aplicatie: [cluster_v11](#) se foloseste o varabila separata pentru alfa



Aplicatia permite stabilirea momentului de afizare precum si reglarea ferestrei de timp.

In diagrama bloc se observa folosirea structurii de date care permite facilitatile amintite.:



Dupa acelasi principiu este realizata si aplicatia: [cluster v12](#).

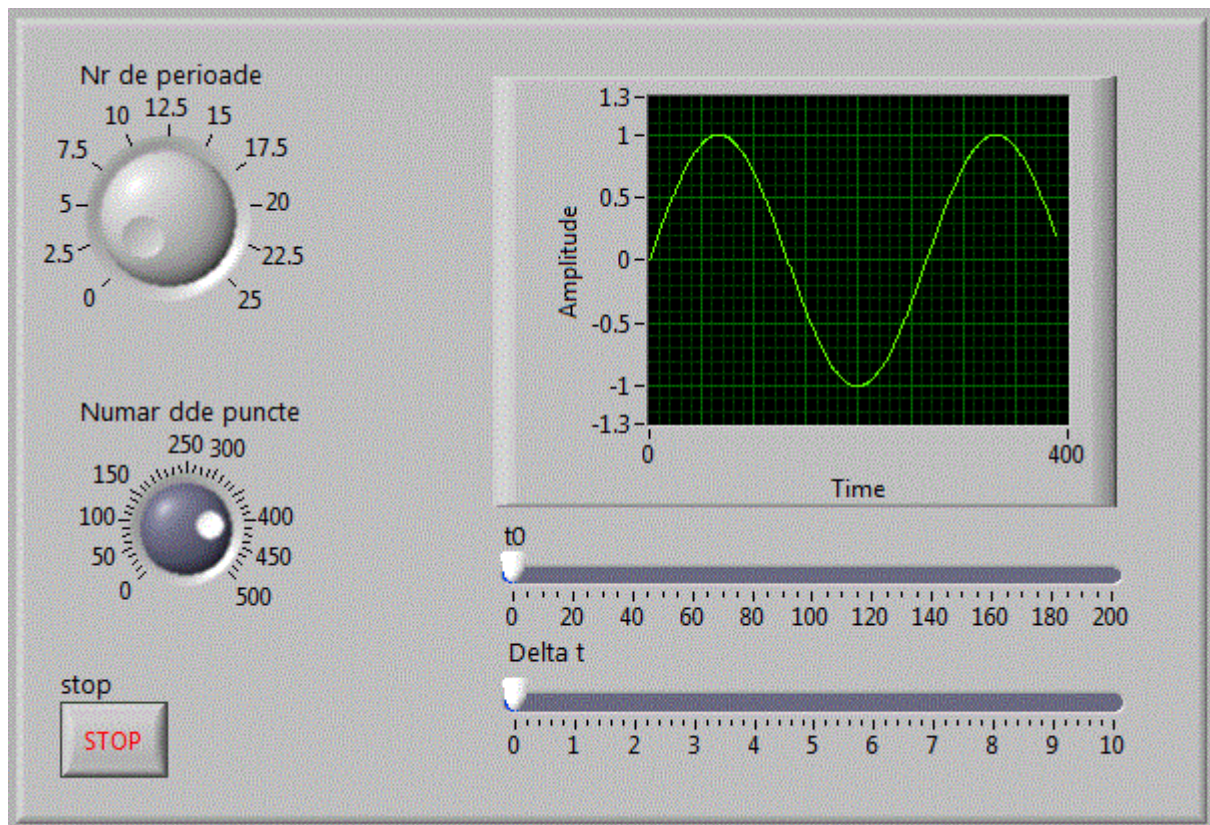
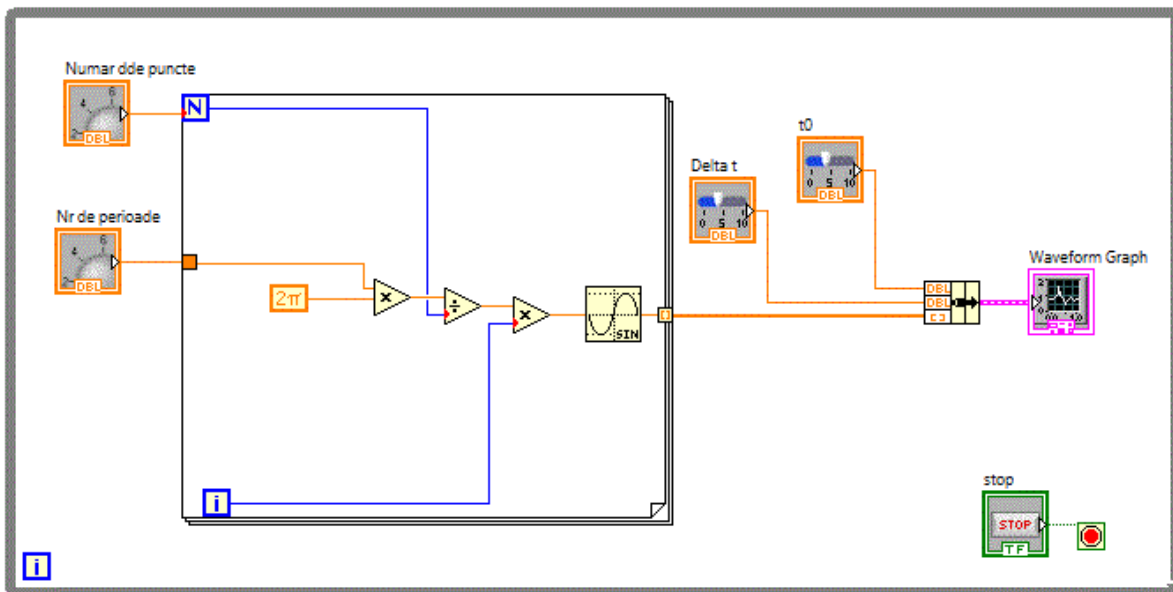


Diagrama bloc fiind:



Putem adauga un zgomot in aplicatia: [cluster v12 n](#) prin adaugarea unui generator de zgomot din:
 Programming-->Signal Processing -->Sig Generation-->Gaussian Noise.

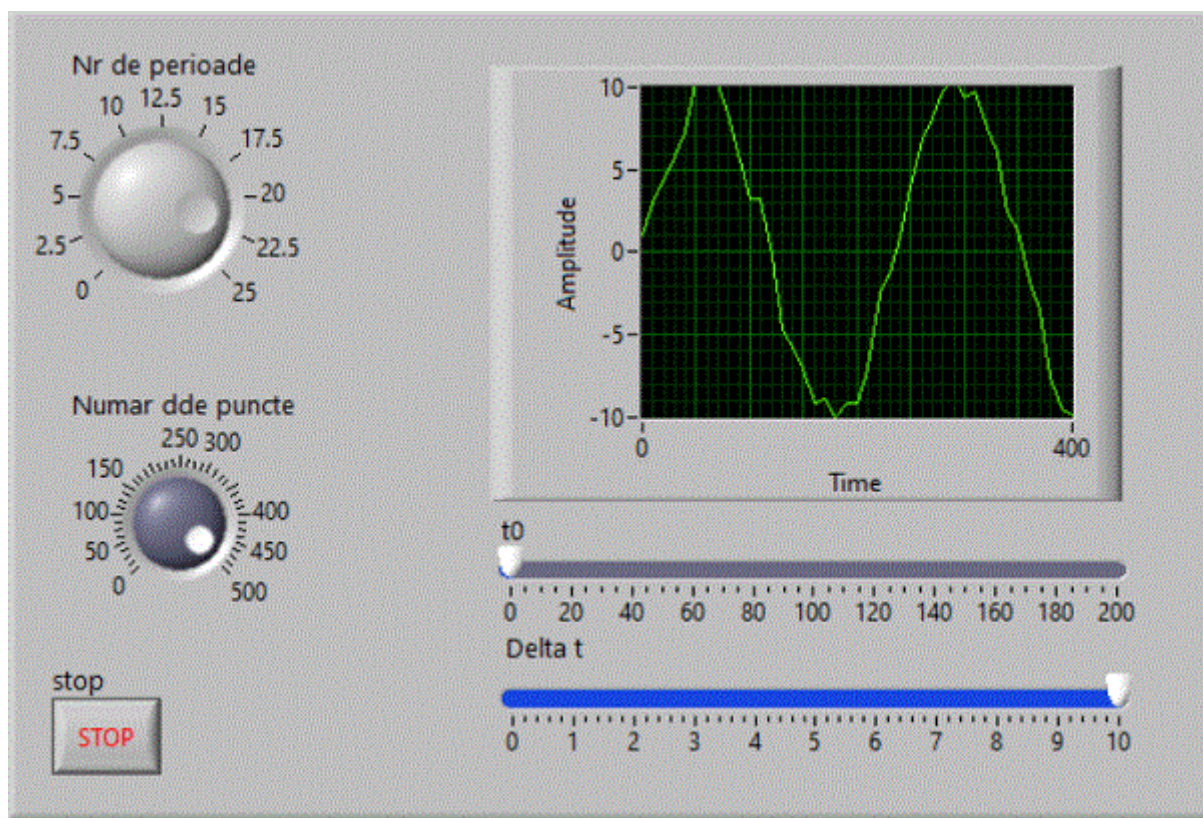
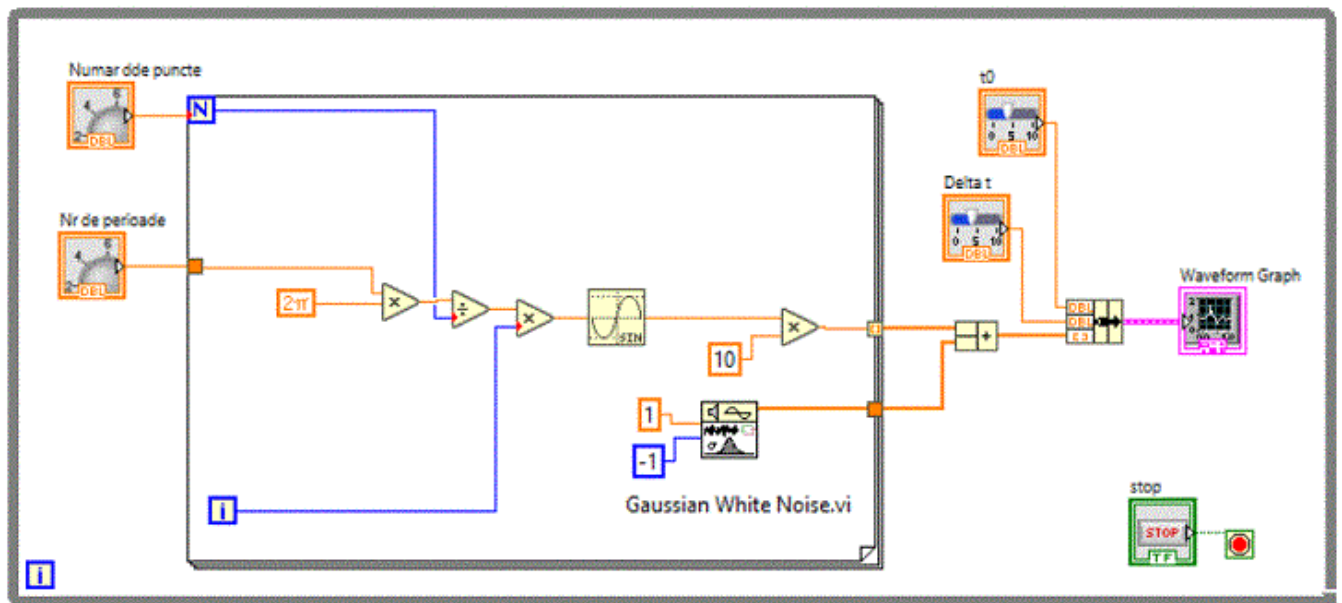


Diagrama bloc fiind:



- **Afisare proportionala si log**

Combinarea controalelor grafice XY Graph si Waveform Graph ne permit afisarea graficului unei functii simultan atat proportional cat si logaritmic. [cluster_v13](#).

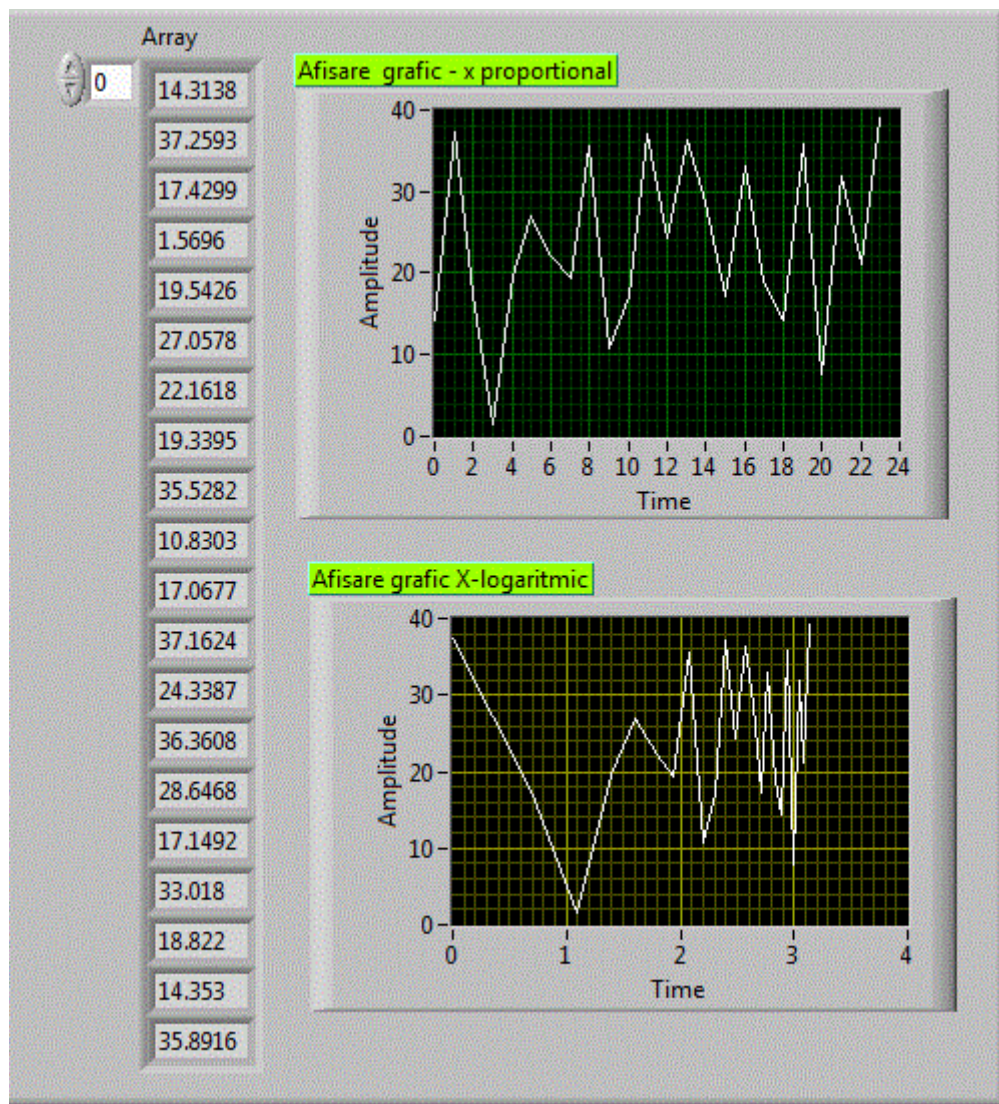
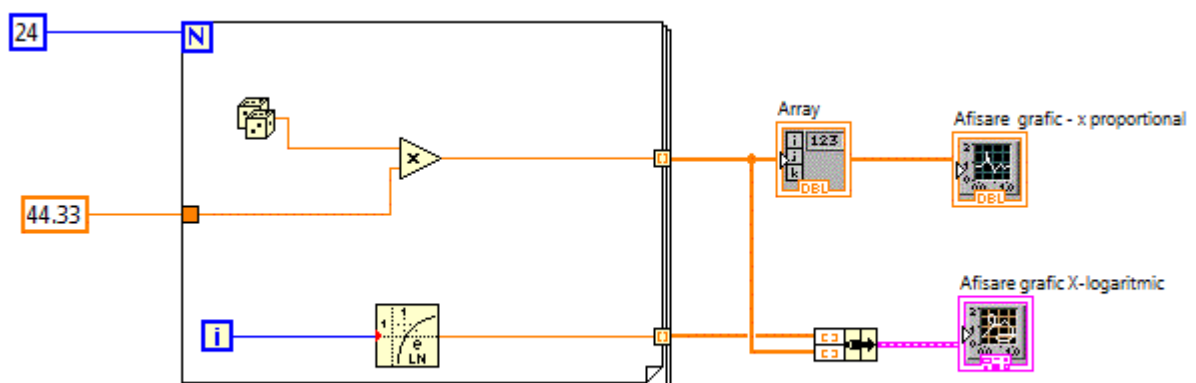
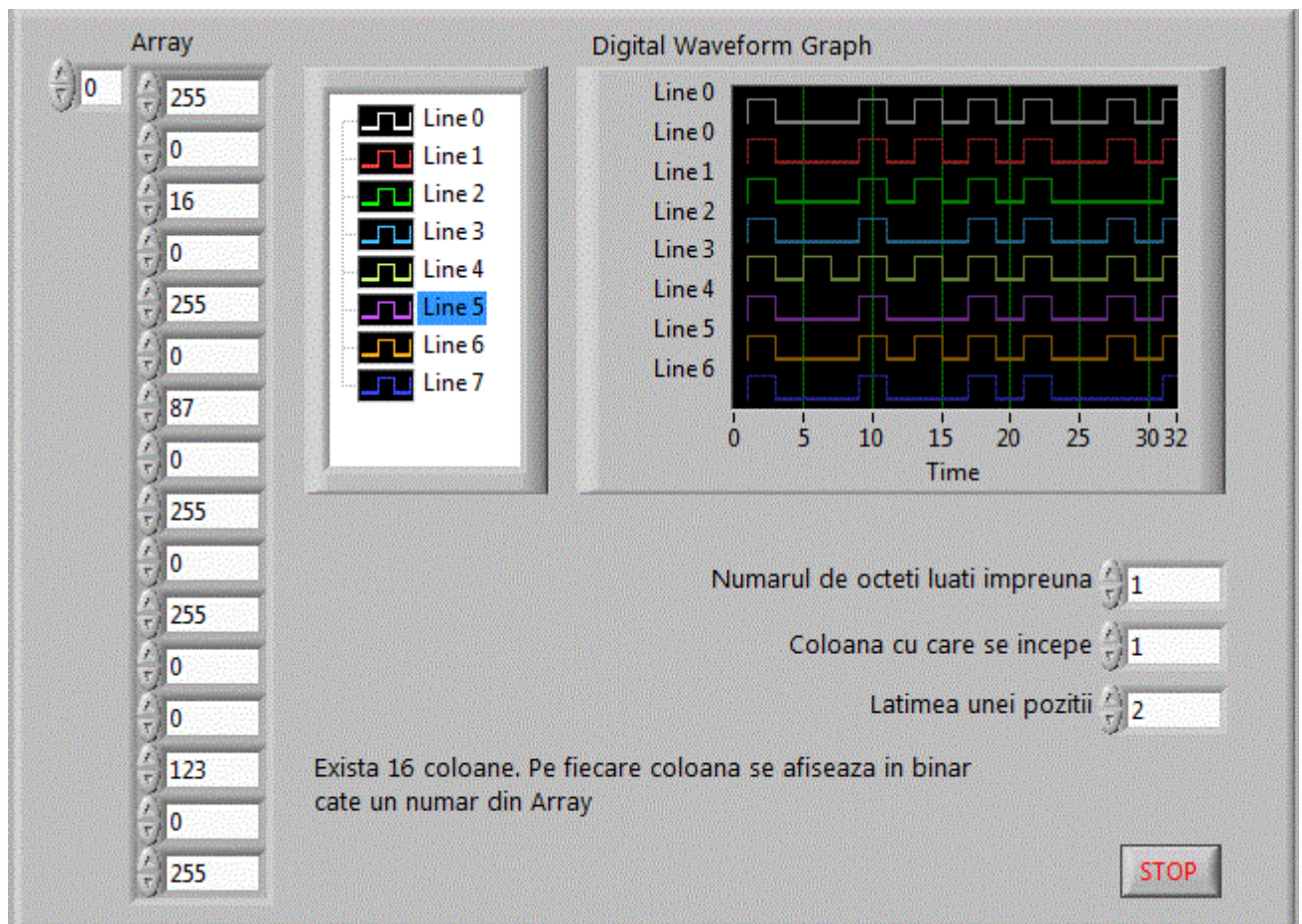


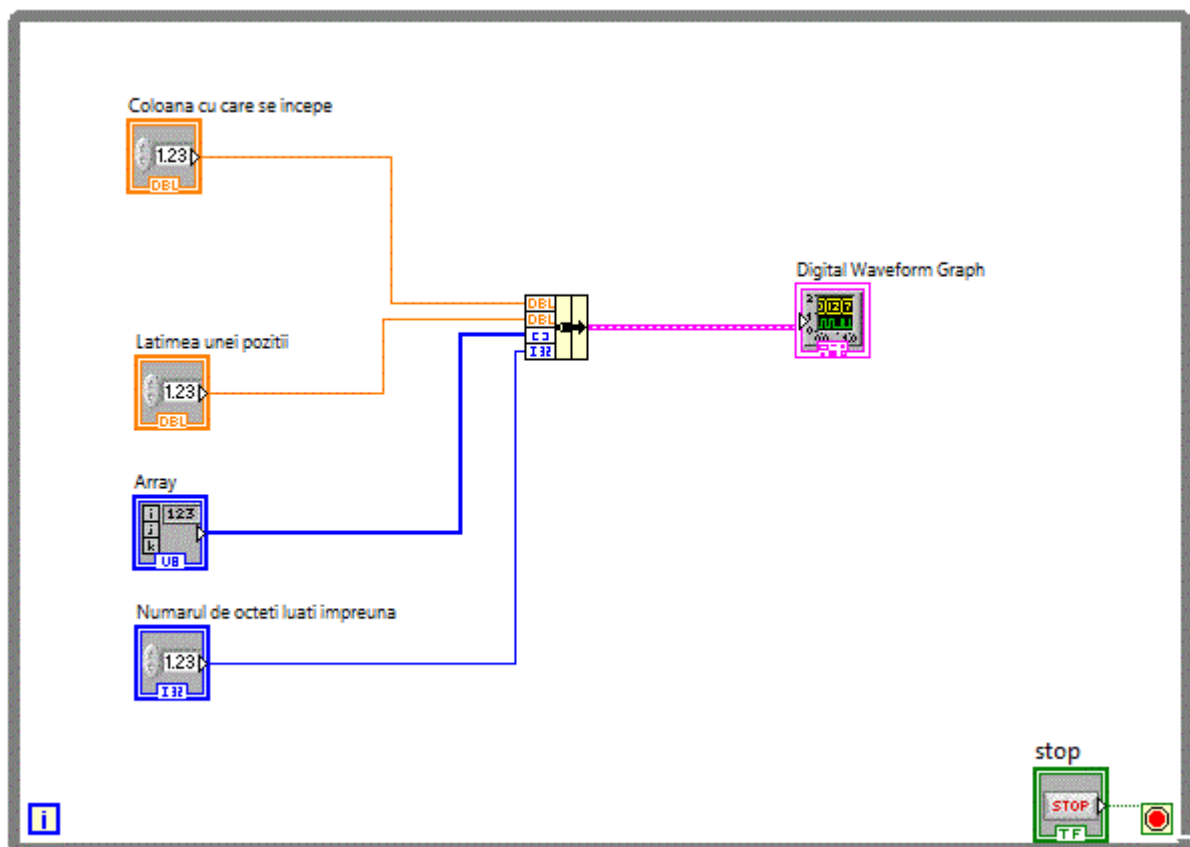
Diagrama bloc fiind:



- **Afisare semnale digitale**

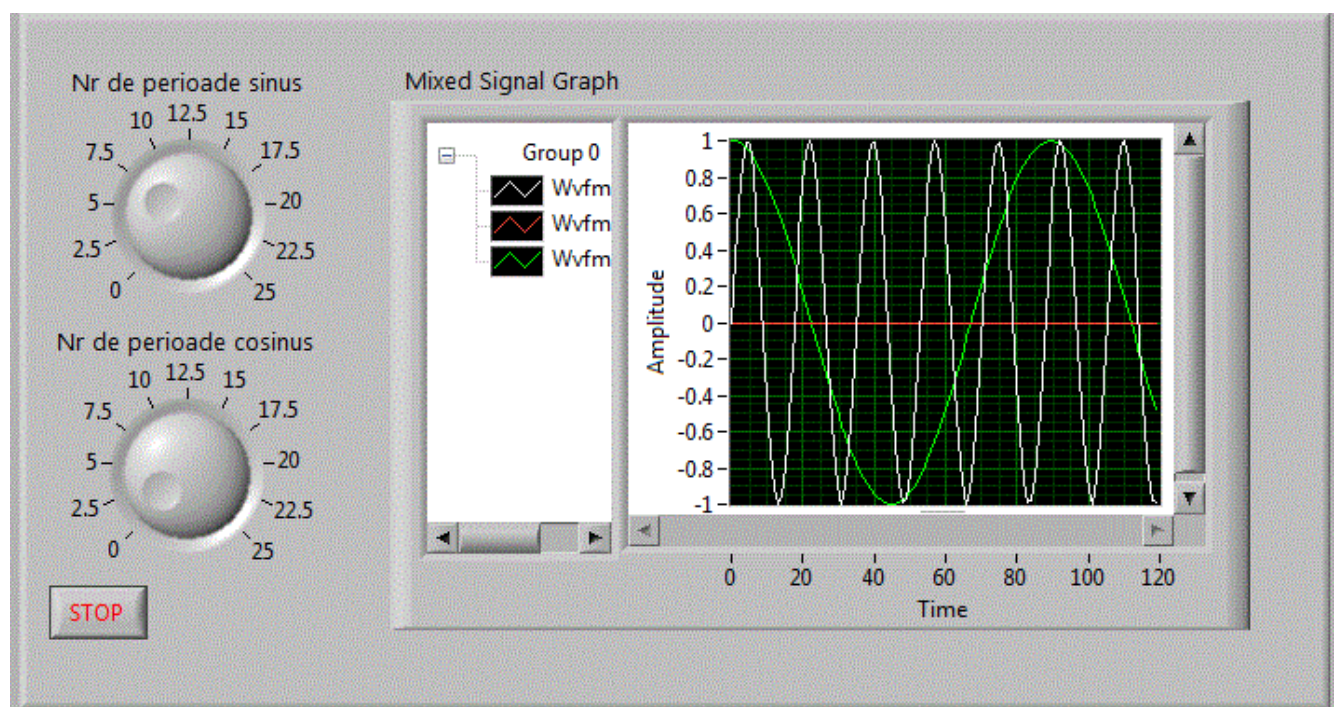
Controlul grafic Digital Waveform Graph este utilizat pentru afisarea valorilor digitale sub forma de forma de unda digitale [cluster v14](#)

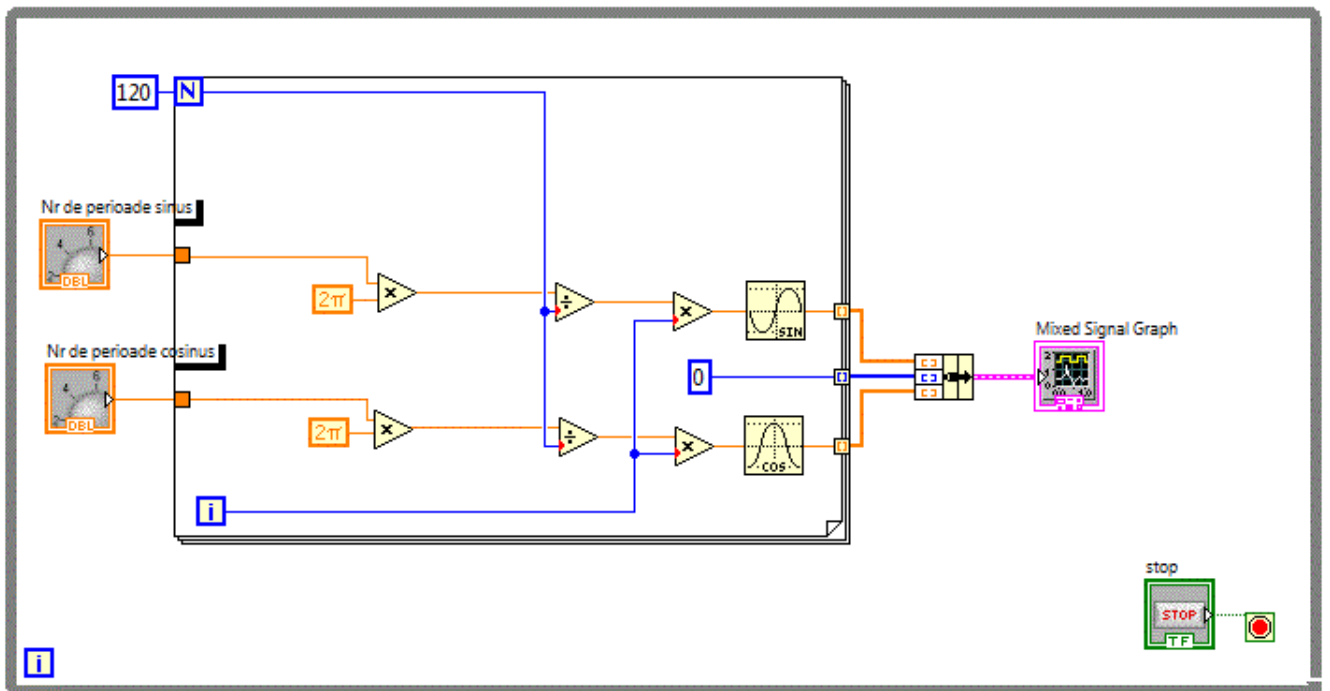




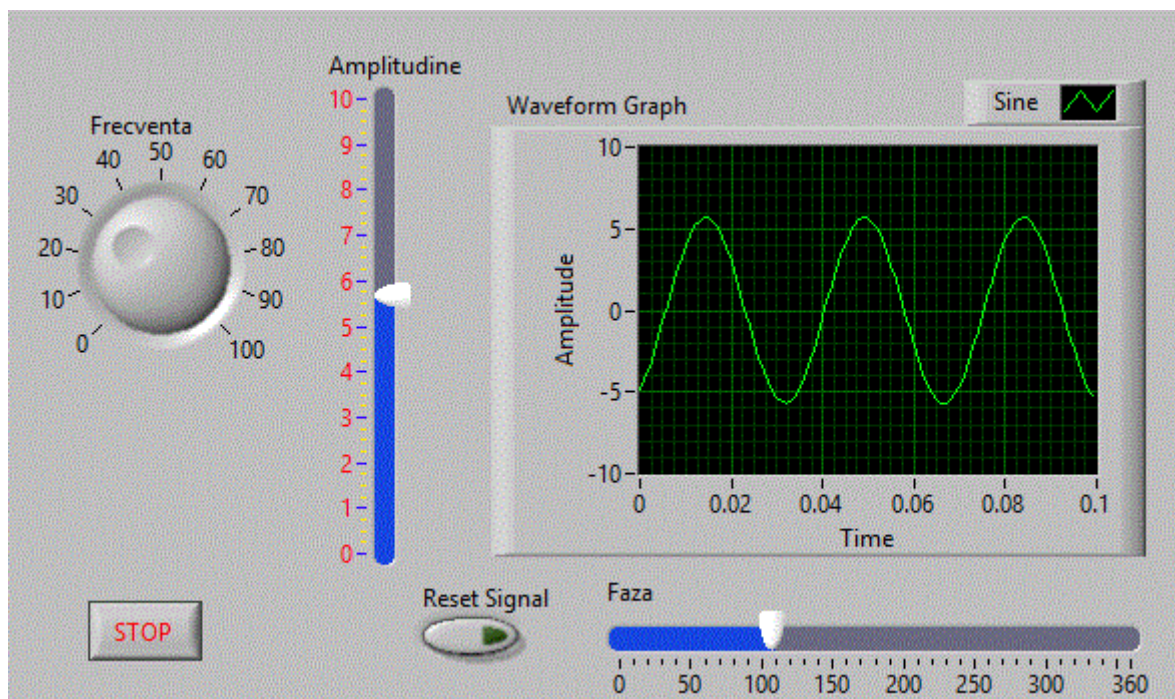
- **Afisare multiplexata pentru mai multe semnale simultan**

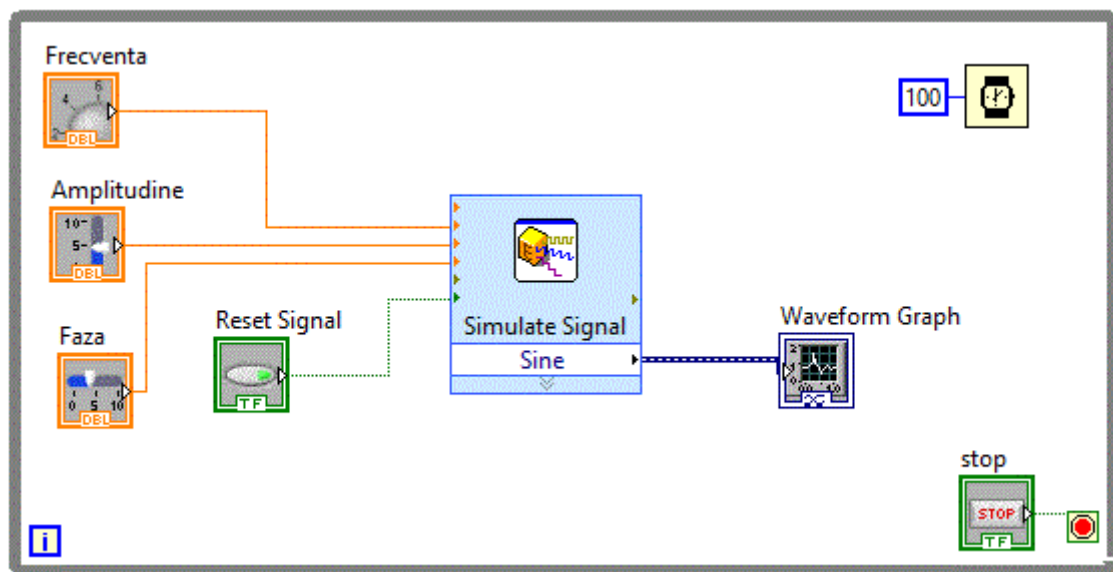
Utilizand controlul Waveform Graph si structuri de date se pot afisa multe semnale simultan. [cluster_v18](#)



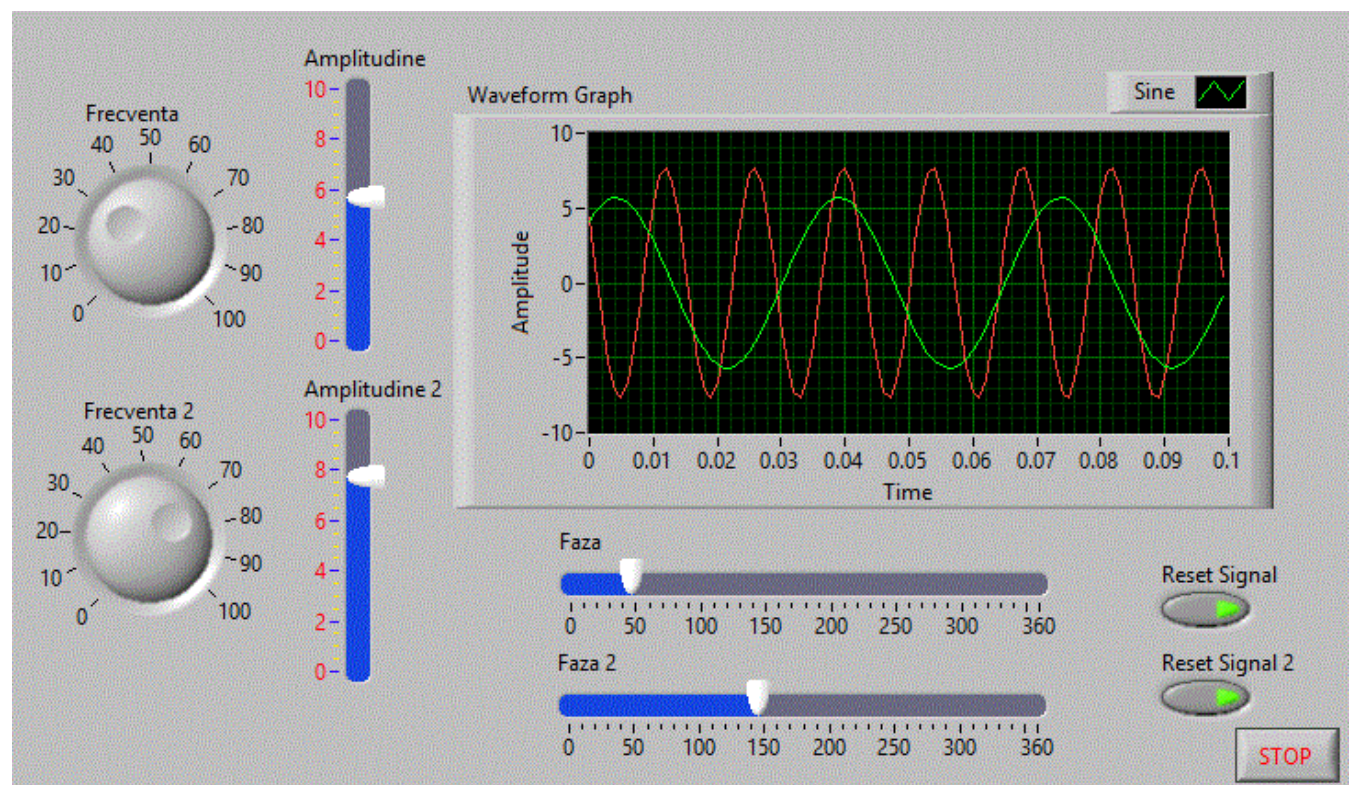


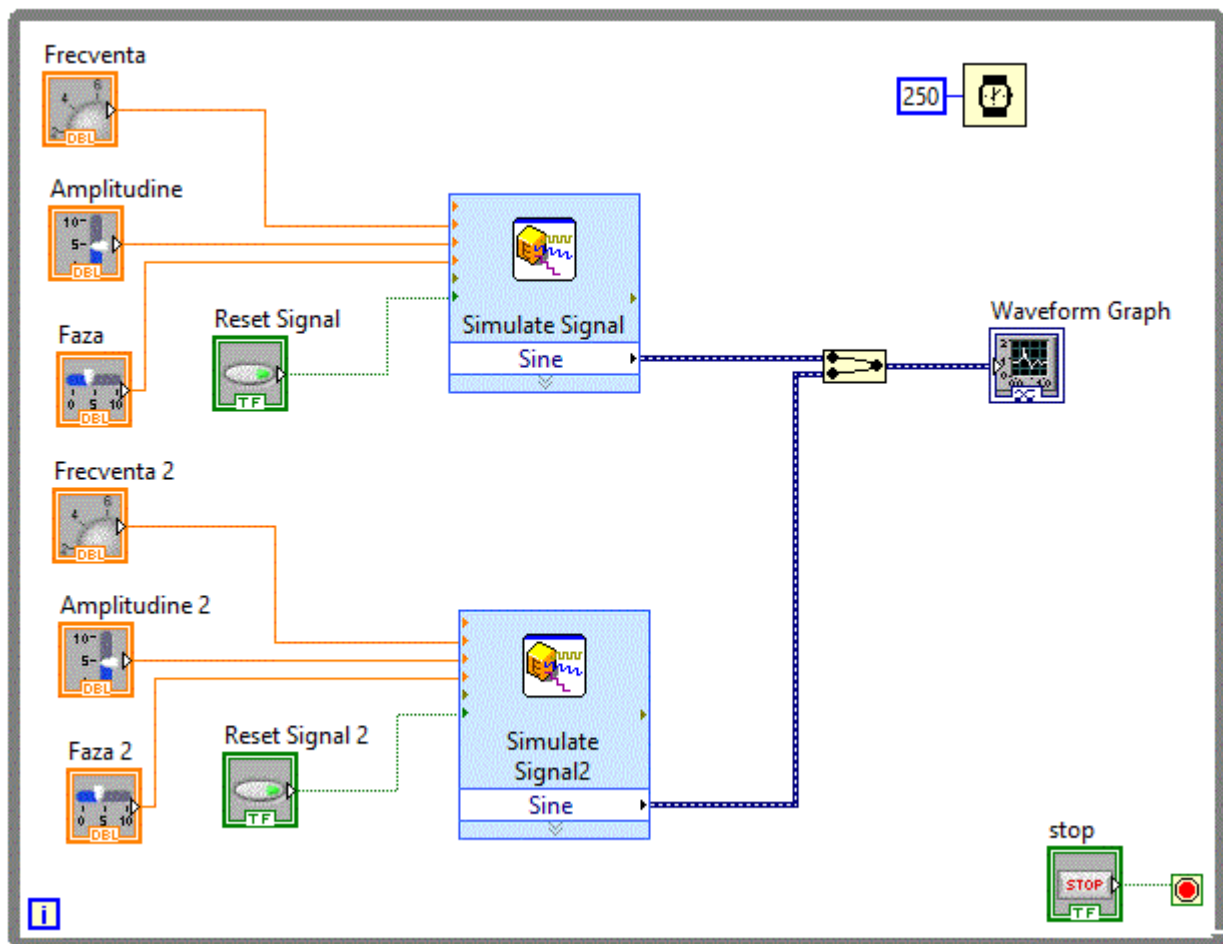
Putem folosi o functie Simulate Sig aflată în: Function->Programming->Waveforms->Analog Wfm->Generation->Simulate Sig care generează diverse forme de undă
[cluster v40](#)





Folosim in continuare functia Simulate Sig pentru a afisa mai multe semnale simultan
[cluster_v41](#)





De data aceasta trebuie sa combinam doua structuri de date, deci nu vom putea folosi "Bundle Function", vom folosi in schimb "Merge Signals Function" pe care o gasim in: Programming-->Express->Sig Manip-->Merge Signal.

