

# Học nhanh LabVIEW

**EVT**Atech Group

Website: <http://dieukhientudong.com>

Contact: hieutq

Dieukhientudong.com

Contact: hieutq | bmnhy2003

EVTAtch Group



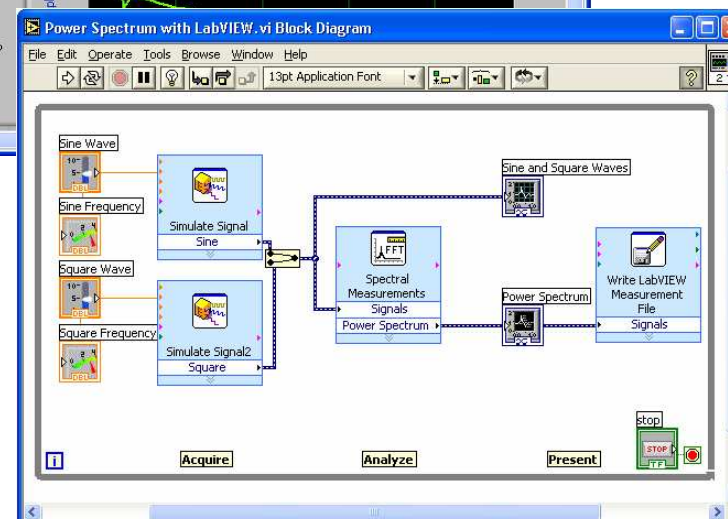
**NI - LabVIEW**  
**EVTAtch**



# LabVIEW™ 8.2

## Lập trình đồ họa dành cho đo lường, điều khiển và kiểm tra.

- Phát triển ứng dụng một cách nhanh chóng với những VI thân thiện và sử dụng môi trường đồ họa đơn giản
- Thiết kế đo lường và điều khiển với giao diện DAQ cho quá trình kết nối với tất cả các kiểu dữ liệu vào/ra.
- Mở rộng với các ứng dụng thời gian thực (Real-Time) cho FPGA và PDA.
- Tài liệu hướng dẫn, trợ giúp với nhiều ngôn ngữ khác nhau.
- Tài liệu TV **EVTatech** đang viết.




# LabVIEW – một ứng dụng hữu ích

- Cuốn sách có tên “Electronic Design” - một phát minh lớn của LabVIEW đã đứng vị trí **Top 50** mốc quan trọng của ngành công nghiệp điện.
- LabVIEW 6.1 đã nhận được nhiều **giải thưởng tự động hoá xuất sắc vào năm 2002**.
- Giải thưởng thiết kế mới với bộ LabVIEW 6i – **một công cụ mang lại hiệu suất cao nhất cho ngành công nghiệp điện năm 2000**
- LabVIEW 6i được lựa chọn là **“Best of the Best”** trong danh sách phần mềm được đánh giá bởi những người trong ngành



# NI LabVIEW: Lịch sử quá trình tăng trưởng

- 
- T5 - 2003 • LabVIEW 7 Express VIs, I/O Assistants, FPGA/PDA targets
  - T1 - 2002 • LabVIEW 6.1 Có khả năng hoạt động mạng network, phân tích
  - T4 - 2000 • LabVIEW 6i Đo lường thông minh, kết nối
  - T3 - 1998 • LabVIEW 5.0 ActiveX, chuyên sâu
  - T2 - 1996 • LabVIEW 4.0 Nhiều công cụ chuyên nghiệp, gỡ rối hoàn thiện
  - T8 - 1993 • LabVIEW 3.0 là phiên bản nền tảng của LabView
  - T9 - 1992 • LabVIEW cho Windows
  - T1 - 1990 • LabVIEW 2.0 cho Macintosh
  - T10 - 1986 • LabVIEW 1.0 cho Macintosh
  - T4 - 1983 • LabVIEW project begins – bắt đầu dự án



# Sản phẩm công nghệ mang tính thương mại

## Phần mềm nhúng vào Labview

- Wolfram Research Mathematica<sup>®</sup>
- Microsoft Excel<sup>®</sup>
- MathWorks MATLAB<sup>®</sup> và Simulink<sup>®</sup>
- MathSoft MathCAD<sup>®</sup>
- Electronic Workbench MultiSim<sup>®</sup>
- Texas Instruments Code Composer Studio<sup>®</sup>
- Ansoft RF circuit design software
- Microsoft Access<sup>®</sup>
- Microsoft SQL Server<sup>®</sup>
- Oracle<sup>®</sup>

## Các giao thức truyền thông

- Ethernet
- CAN
- DeviceNet
- USB
- IEEE 1394
- RS-232
- GPIB
- RS-485



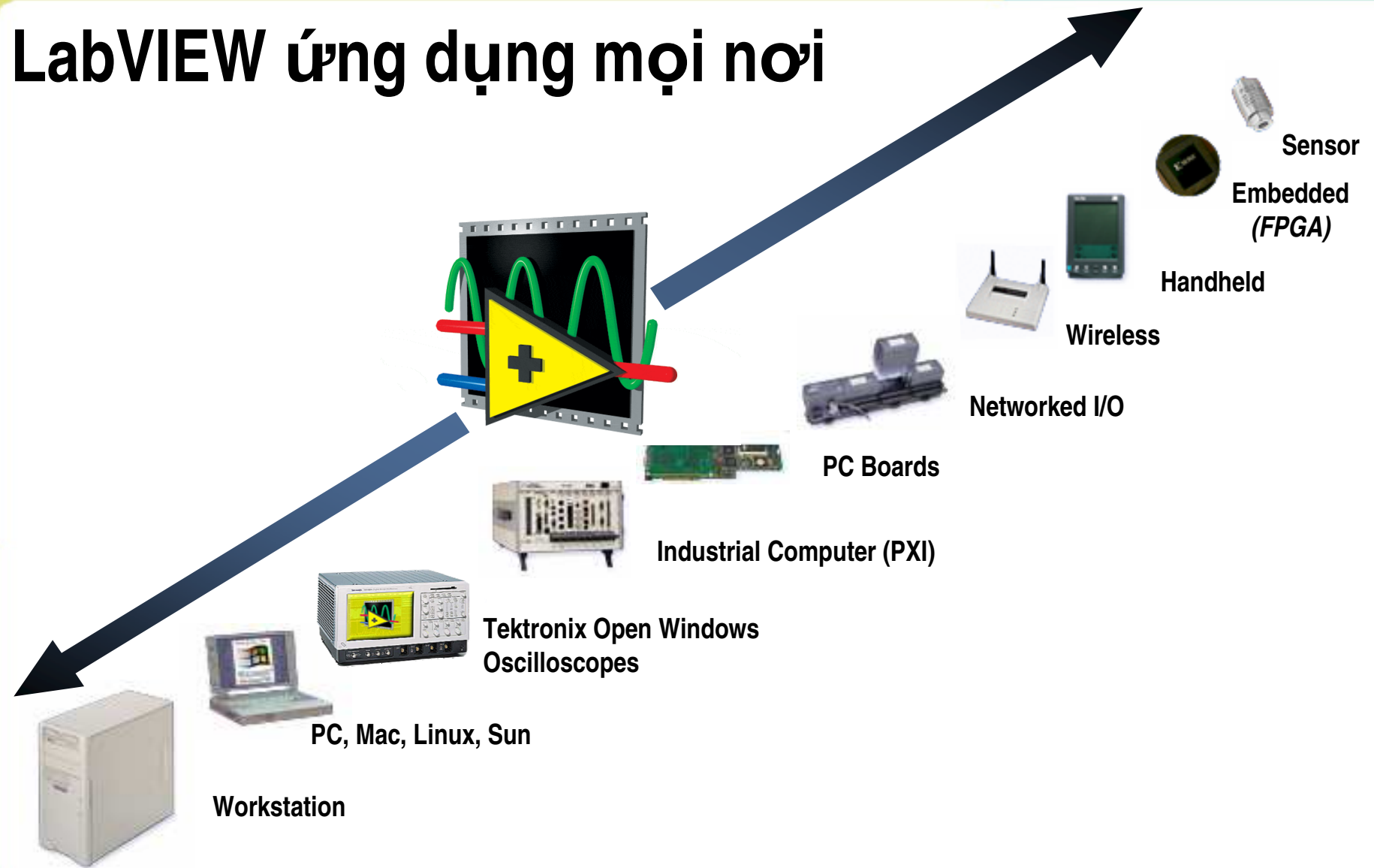
[Dieukhientudong.com](http://Dieukhientudong.com)

Contact: hieutq | bmnhy2003



NI - LabVIEW  
EVTech

# LabVIEW ứng dụng mọi nơi



[Dieukhientudong.com](http://Dieukhientudong.com)

Contact: [hieutq](mailto:hieutq) | [bmnhy2003](mailto:bmnhy2003)



NI - LabVIEW  
EVTAtch

# Họ LabVIEW

## NI LabVIEW

Phần mềm lập trình đồ họa dành cho đo lường và điều khiển

LabVIEW Real-Time Module

LabVIEW FPGA Module

LabVIEW PDA Module

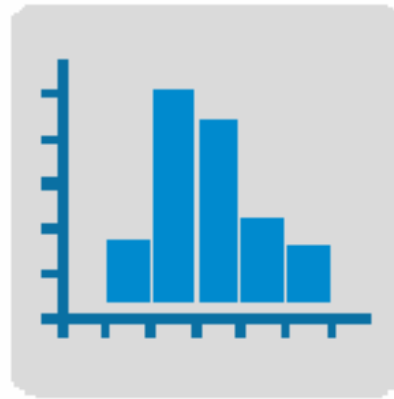
LabVIEW Datalogging and  
Supervisory Control Module

[Dieukhientudong.com](http://Dieukhientudong.com)

Contact: [hieutq](mailto:hieutq) | [bmnhy2003](mailto:bmnhy2003)



**NI - LabVIEW**  
**EVTAtch**



## *Cái có được, phân tích và hữu hiệu*

LabVIEW là một ngôn ngữ lập trình đồ họa khá mạnh trong các lĩnh vực kiểm tra, đo lường, và điều khiển. Có thể thấy rõ trong 3 điểm nổi bật sau: **cái có được, phân tích và sự hữu hiệu**. LabVIEW là một phần mềm thân thiện, một công cụ mạnh cho phân tích và hữu dụng trong lập trình thời gian thực giới thực.





# Cái có được ở LabVIEW



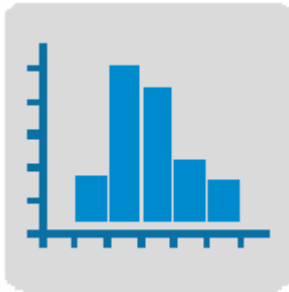
LabVIEW được kết hợp chặt chẽ với phần cứng, thêm vào hàng ngàn kết nối với các thiết bị vào/ra với hàng trăm thiết bị khác nhau.

**LabVIEW có thể được sử dụng dành cho nhiều thiết bị:**

- GPIB, Serial, Ethernet, VXI, PXI Instruments
- Data Acquisition (DAQ)
- PCI eXtensions for Instrumentation (PXI)
- Image Acquisition (IMAQ)
- Motion Control
- Real-Time (RT) PXI
- PLC (through OPC Server)
- PDA
- Modular Instruments



# Phân tích ở LabVIEW



**Hệ thống phân tích đo lường mạnh được xây dựng trên môi trường phát triển Labview.**

**LabVIEW bao gồm nhiều các công cụ trợ giúp người dùng để phân tích dữ liệu:**

- Hơn 400 mẫu hàm phân tích đo lường trong các biểu thức khác, tối ưu, lọc, toán học, chuỗi số học, thống kê, v.v...
- 12 VI Express mới đặc biệt được thiết kế dành cho phân tích đo lường, bao gồm bộ lọc và phân tích quan phổ.
- Các VI xử lý tín hiệu dành cho Filtering, Windowing, Transforms, Peak Detection, Harmonic Analysis, Spectrum Analysis, v.v.

# Sự hữu hiệu ở LabVIEW



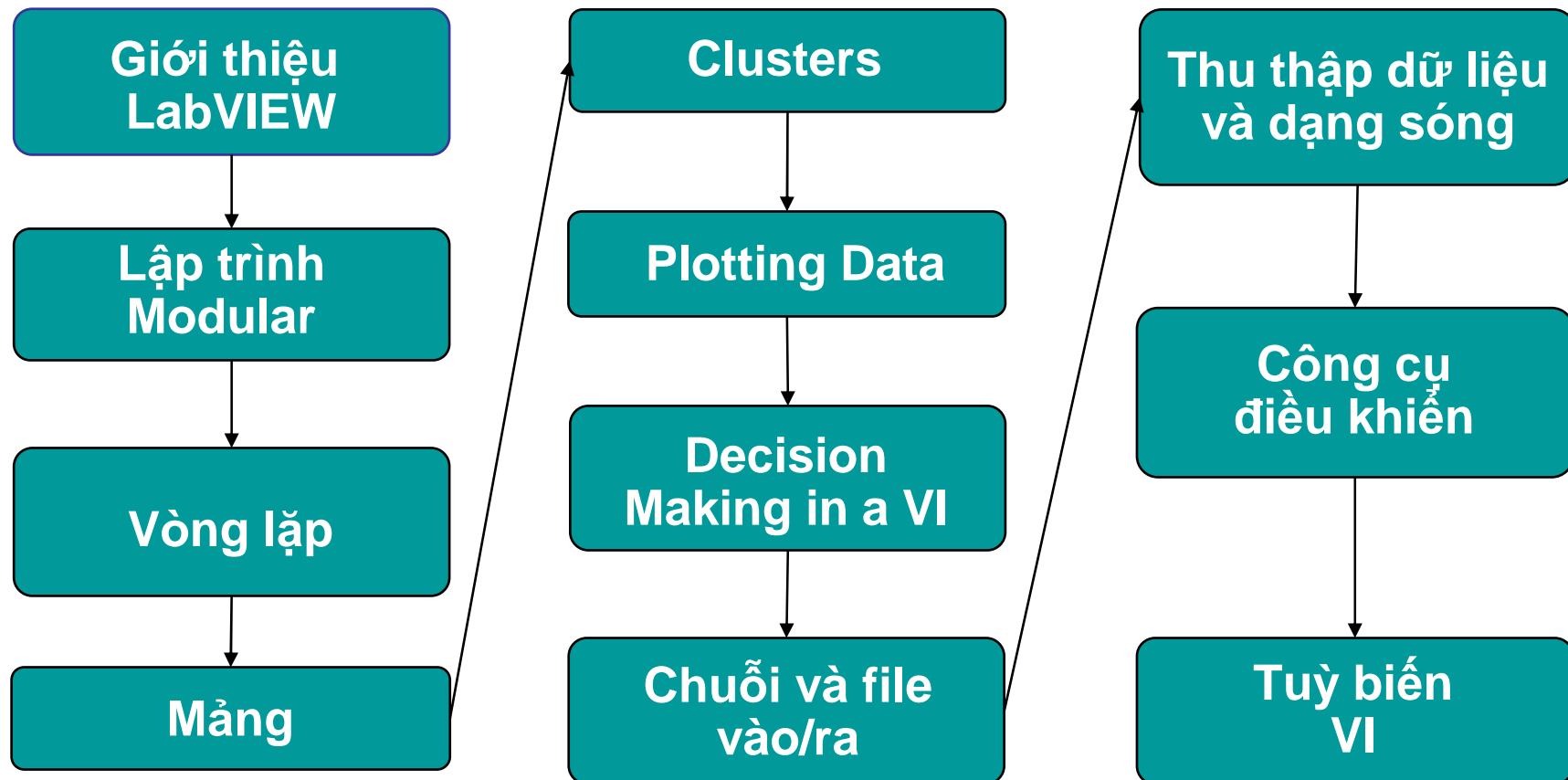
**LabVIEW có thể làm việc trên PC của bạn hoặc ngay cả trên mạng network, hoặc bạn có thể thêm vào những ứng dụng là 1 lợi thế giống như DIAdem.**

**LabVIEW includes the following tools to help you present your data:**

- On your machine — Graphs, Charts, Tables, Gauges, Meters, Tanks, 3D Controls, Picture Control, 3D Graphs (Windows Only), Report Generation (Windows Only)
- Over the Internet — Web Publishing Tools, Datasocket (Windows Only), TCP/IP, VI Server, Remote Panels, Email
- Enterprise Connectivity Toolset — SQL Tools (Databases), Internet Tools (FTP, Telnet, HTML)



# Các bài sẽ giới thiệu về Labview cuốn này



# Điều đạt được của khoá học

## Khoá học này chuẩn bị cho bạn:

- Hiểu được Front panels, biểu đồ thành lập các khối thành một hệ thống, và các thiết bị kết nối.
- Biết sử dụng cấu trúc các chương trình và các dạng dữ liệu mà chúng tồn tại ở LabVIEW
- Biết sử dụng linh hoạt cách sắp xếp dữ liệu cho việc lập trình máy tính và kỹ thuật xác định, sửa các lỗi trong một chương trình.
- Tạo ra và lưu trữ các VI của bạn vì vậy bạn có thể dùng chúng như những VI phụ trợ.
- Hiển thị và lưu trữ dữ liệu của bạn.
- Tạo ra những ứng dụng sử dụng những bảng plug-in data acquisition (DAQ)
- Tạo ra các ứng dụng mà sử dụng GPIB và chuỗi các thiết bị cổng.



# Khoá học có thể mang lại.

**Mục đích của khoá học đó không phải là để bàn luận về những vấn đề như là:**

- Xây dựng VI trong LabVIEW, chức năng, hoặc thư viện VI
- Thuyết tương tự và công nghệ.
- Hoạt động chi tiết của chuỗi cổng hoặc bus GPIB
- Làm thế nào để phát triển một công cụ điều khiển.



# Bài học 1:

## Giới thiệu về LabVIEW

### Chủ đề nghiên cứu

- Môi trường LabVIEW
- Giao diện Labview
- Biểu đồ thiết lập các khối thành một hệ thống
- Lập trình
- Trợ giúp trong LabVIEW
- Gỡ rối 1 VI



# Virtual Instruments (VIs)

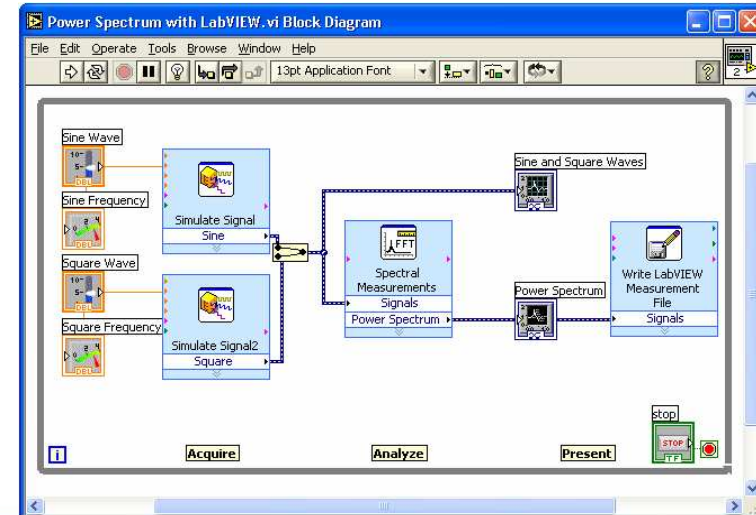
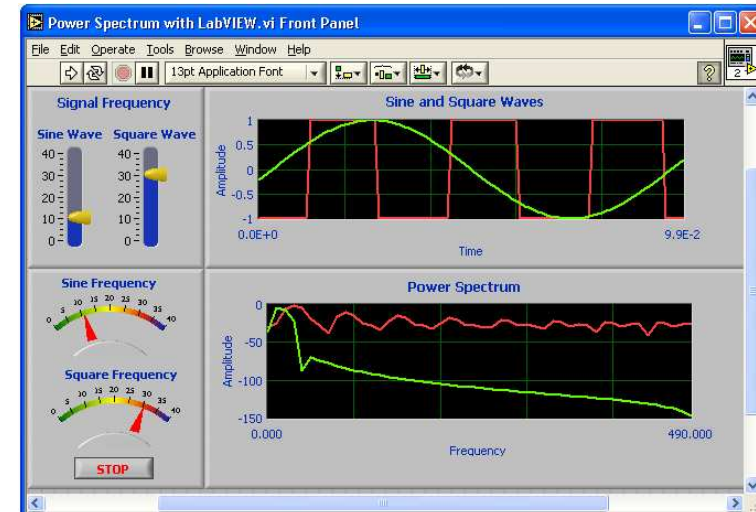
## – Những công cụ ảo

Giao diện chính

- Controls = Inputs (Vào)
- Indicators = Outputs (Ra)

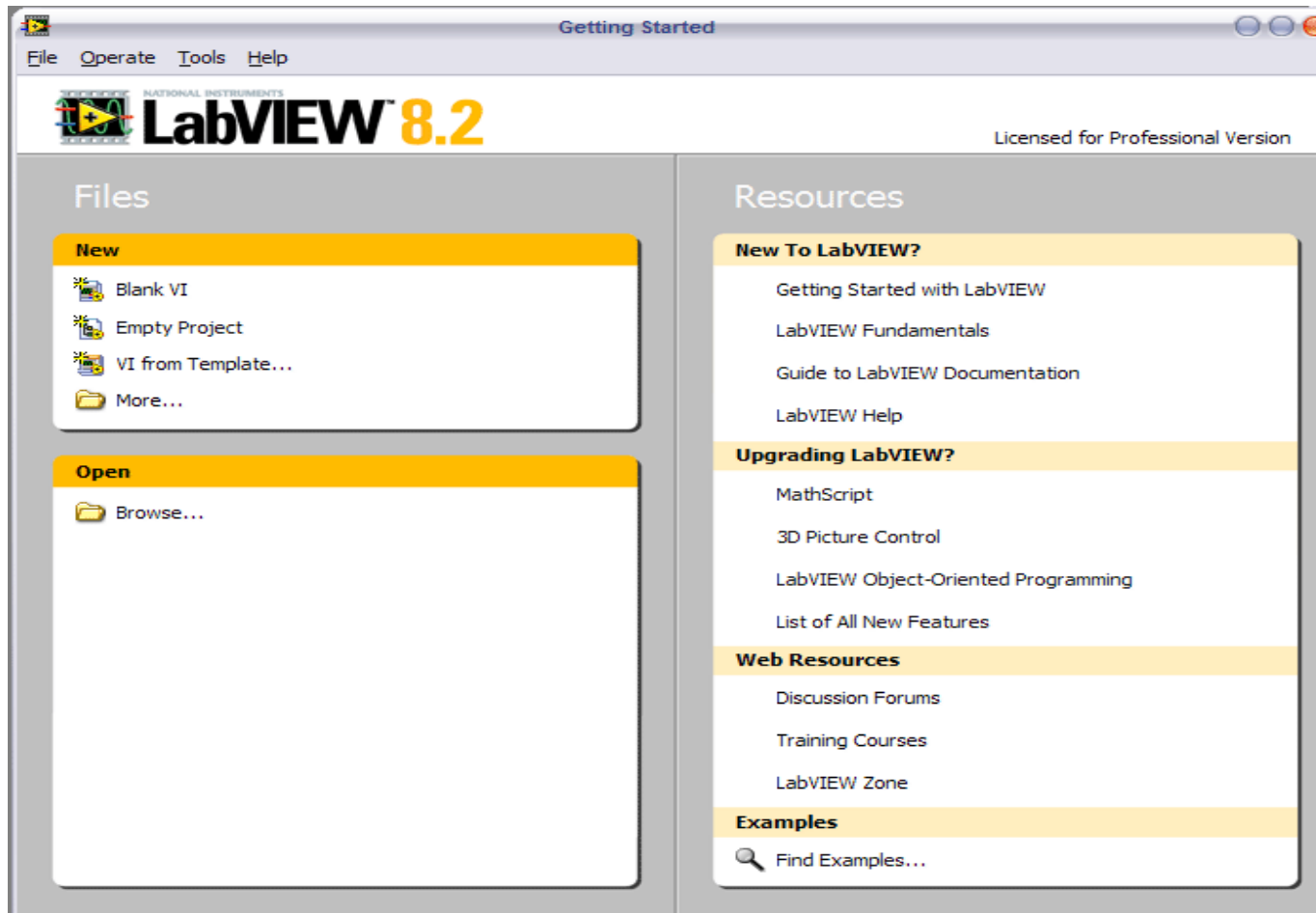
Biểu đồ khối

- Các khối chương trình của giao diện chính
- Các thành phần đi dây.

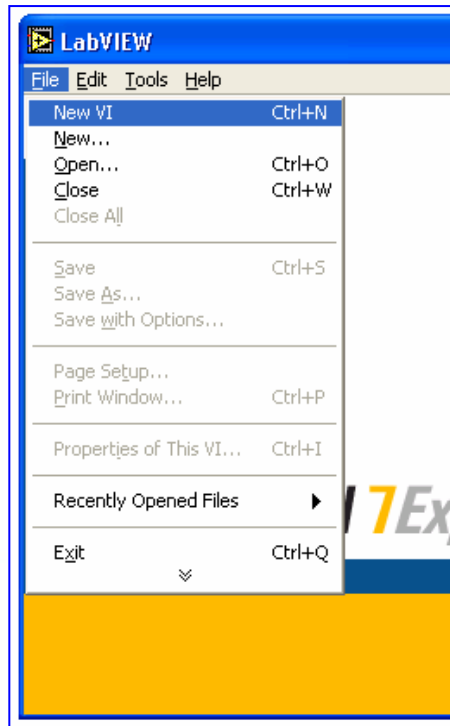




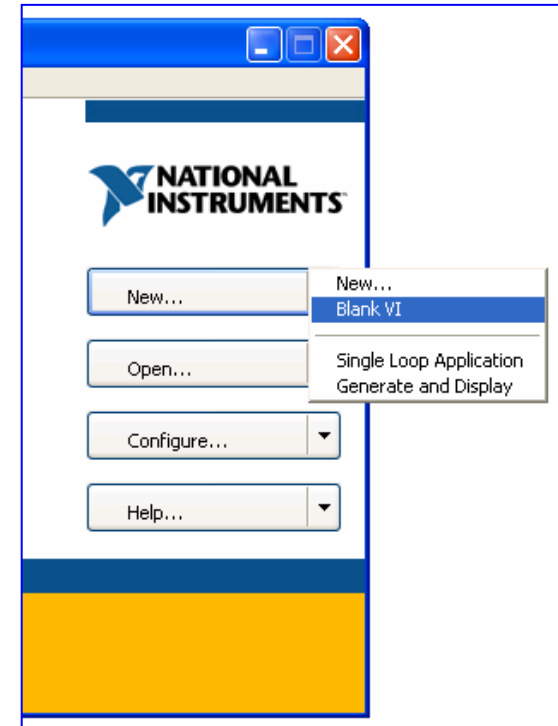
# Hộp thoại khởi động LabVIEW



# Tạo một VI mới

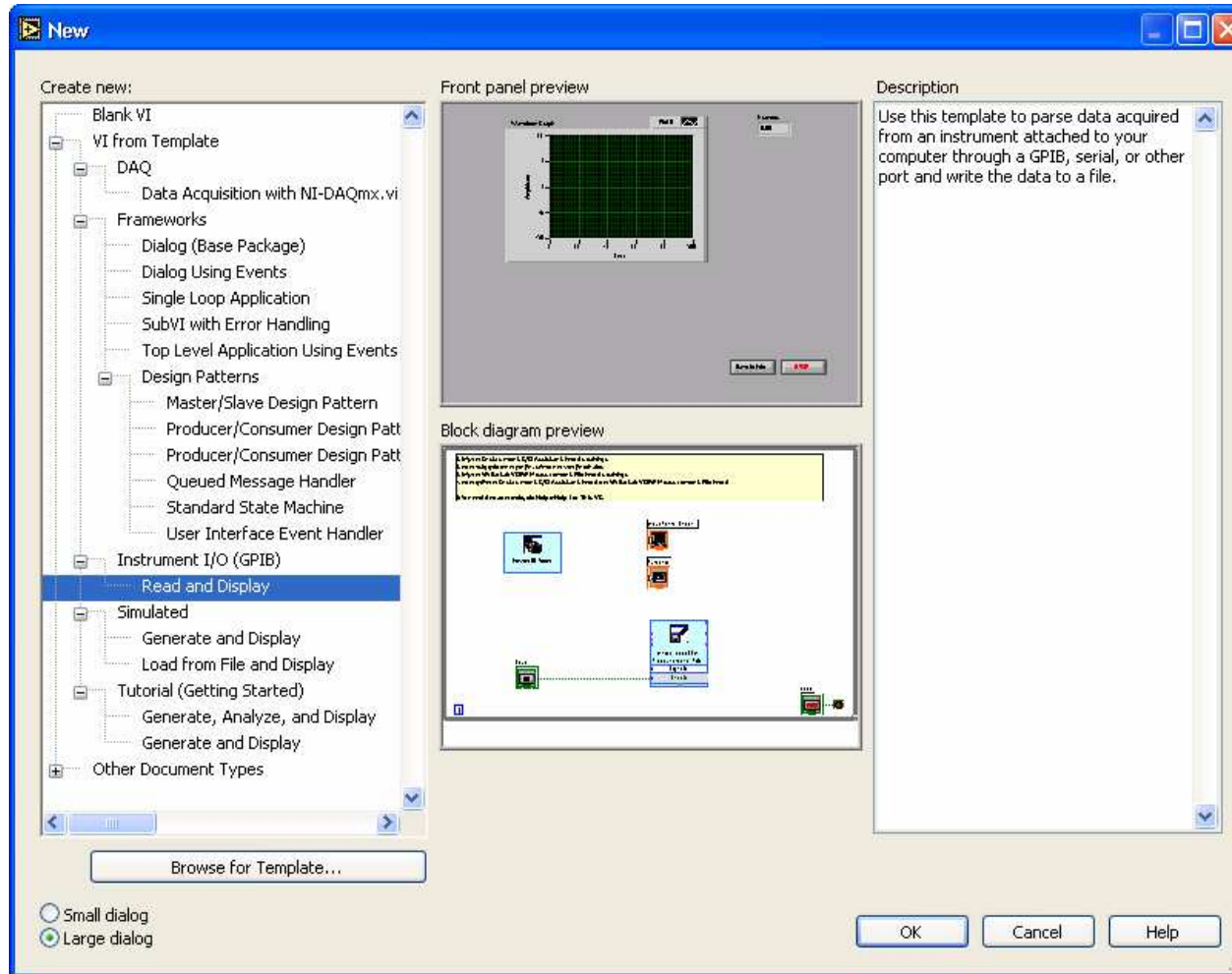


- **File»New VI** để tạo 1 VI mới



- **File»New...** Để mở một hộp thoại mới và cấu hình cho một VI, đặt các biến số, điều khiển, v.v...

# Mở các mẫu có sẵn



# Menu

File Edit Operate Tools Browse Window Help

Di chuyển và click đến thực đơn trên Menu để lựa chọn theo các yêu cầu của bạn



# Giao diện phía trước

Thanh công cụ

Nút điều khiển

Nhãn đồ thị sóng

Đồ thị dạng sóng

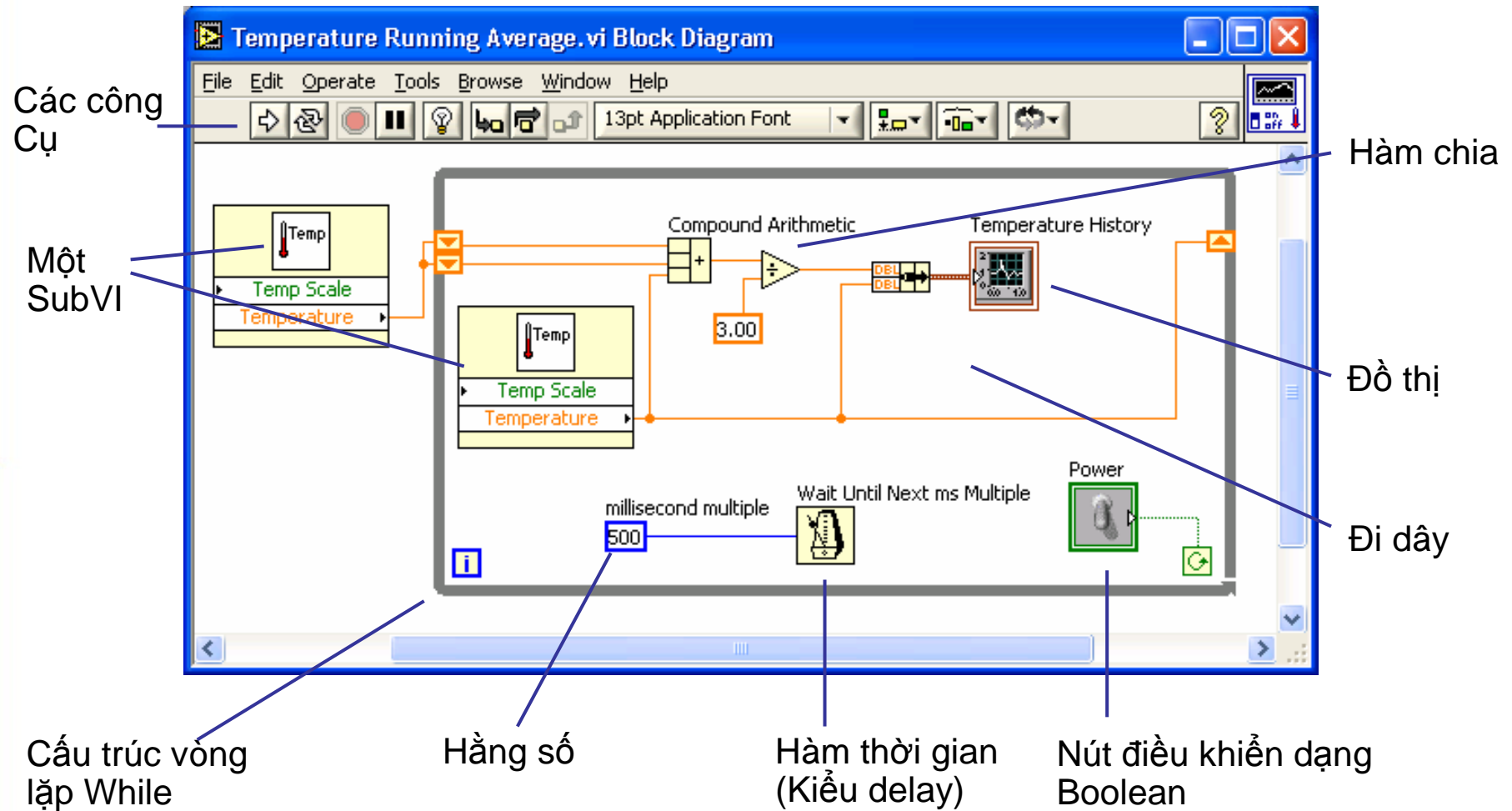
Chú thích Plot

Biểu tượng

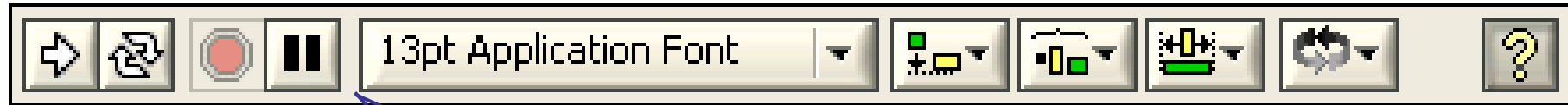
Chú thích đồ thị

Chú thích Scale

# Đồ thị khối



# Các thanh công cụ và biểu đồ khối



Chạy ứng dụng

Chạy ứng dụng liên tục

Ngừng chạy

Tạm dừng/Tiếp tục



Những nút được thêm vào trên thanh công cụ

- Hiện thị quá trình thi hành
- Nút Step Into
- Nút Step Over
- Nút Step Out

Font ring

Alignment ring

Distribution ring

Resize ring

Reorder ring

Trợ giúp ngữ cảnh



Chỉ dẫn



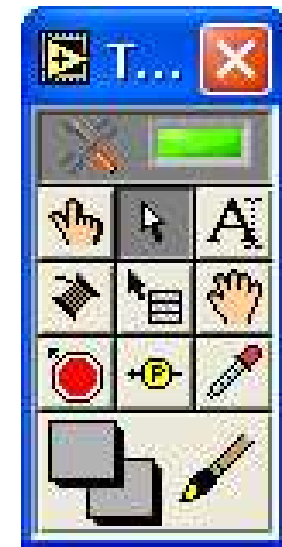
Chấp nhận



Chạy dứt quãng

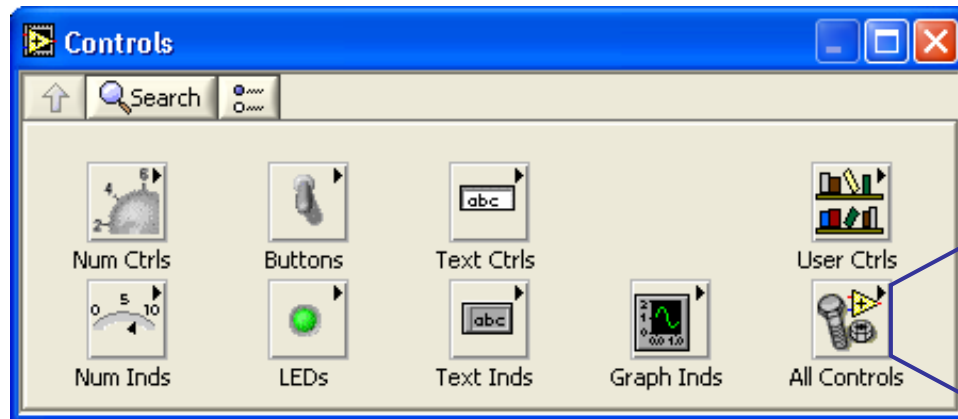
# Bảng công cụ

- LabVIEW tự động chọn những công cụ cần thiết.
- Hiện thị trên Front panel và trên biểu đồ thiết lập hệ thống.
- Các chức năng công cụ có thể lựa chọn ở chế độ đặc biệt theo con trỏ chuột.
- Sử dụng các công cụ để tạo và chỉnh sửa giao diện chính và các đối tượng của biểu đồ khối
- Hiện thị bảng công cụ (Tools Palette), chọn **Window»Show Tools Palette**





# Giao diện chính - Bảng điều khiển (Controls Palette)



## Controls Palette

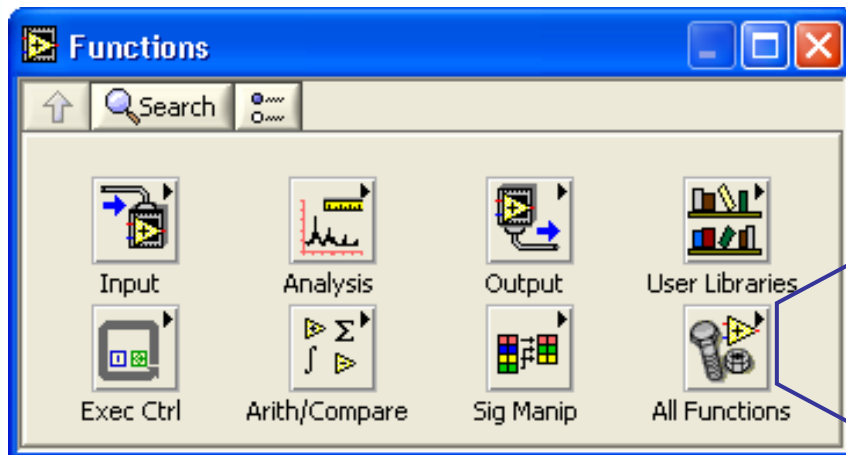
Bao gồm hầu hết các hàm phổ biến nhất được sử dụng để điều khiển



## Bảng Controls

Hiển thị các chức năng điều khiển

# Biểu đồ khối- Bảng các hàm (Functions Palette)



## Functions Palette

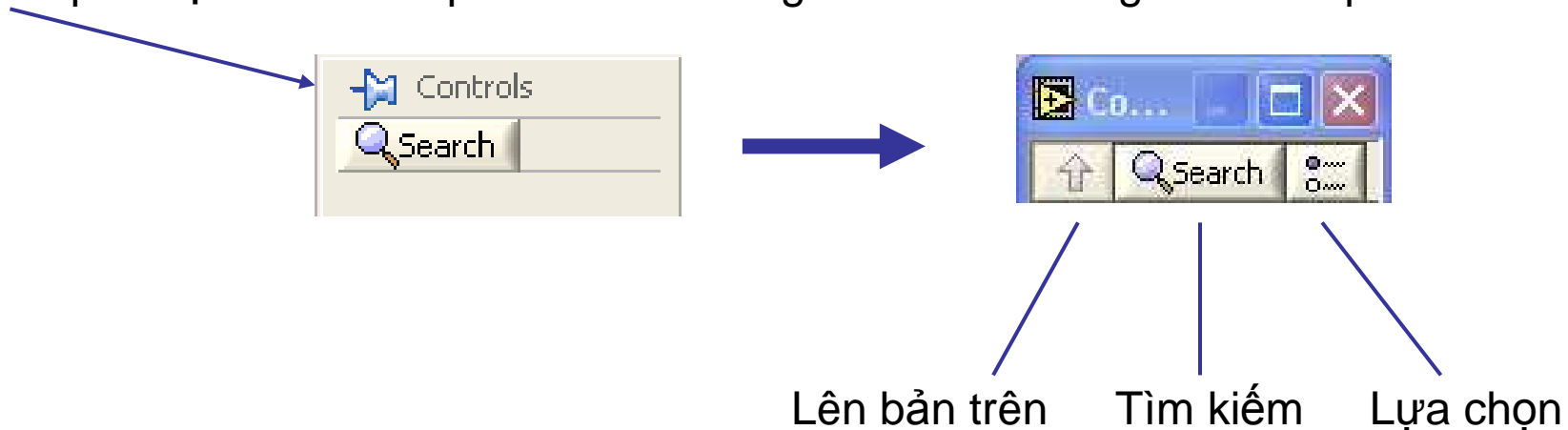
Bao gồm các VI (Các VI tương tác với trang cấu hình) và hầu hết các thành phần phổ biến của các hàm.



**Bảng Functions**  
Hiện thị tất cả các hàm

# Bảng các công cụ

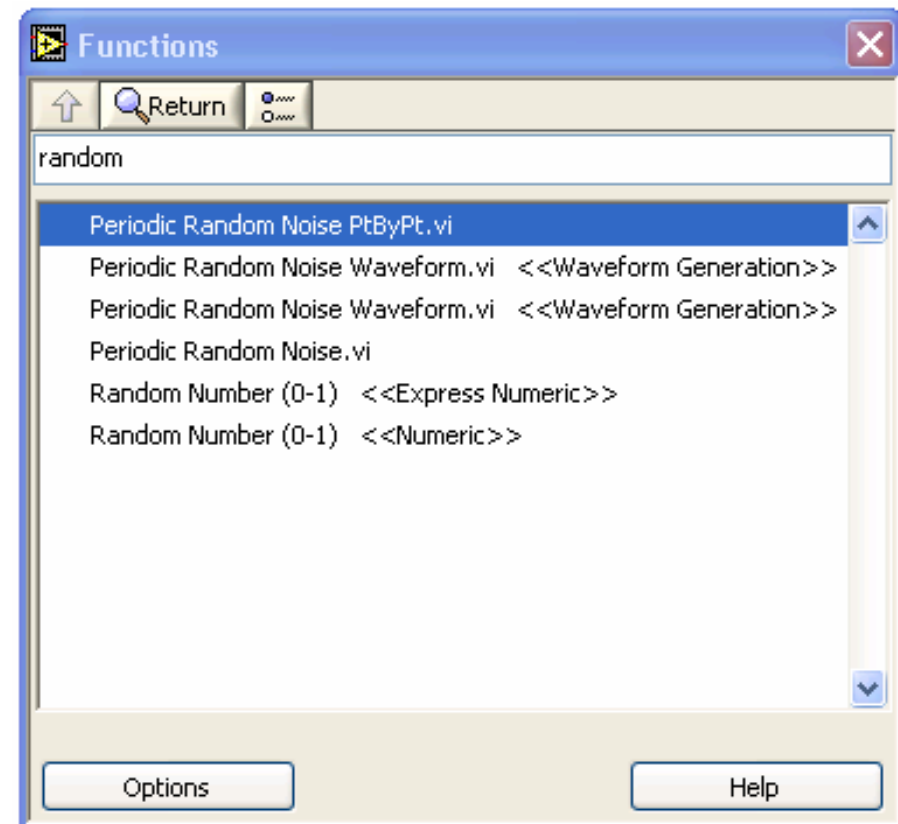
Nhấp chuột vào thành phần VI cần dùng để đưa tới bảng các VI tiếp theo



- Là các bảng dạng đồ họa, luân chuyển
- Bảng cấp 2 được luân chuyển từ bảng chính
- Sử dụng bảng lựa chọn để thay đổi tổng quan từ lựa chọn nhanh sang chuyên nghiệp

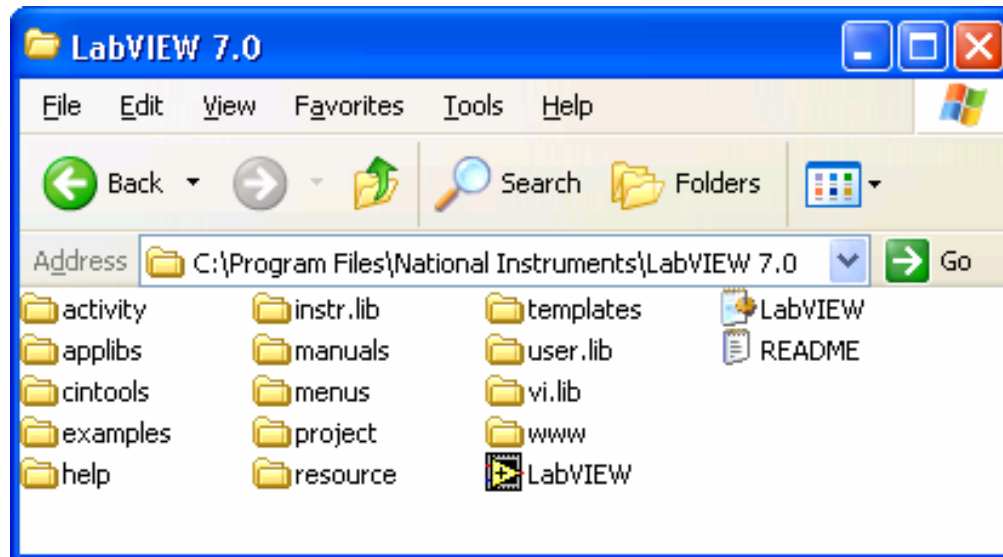
# Tìm kiếm một Controls, VIs, và Functions

- Nhấp vào nút search và gõ chữ bạn muốn tìm.
- Nhấp chuột và kéo thả mọi thành phần từ cửa sổ tìm kiếm tới biểu đồ khối hoặc nhấp đúp chuột vào thành phần đó để mở



# Bảng Control & Function theo người dùng

Vào: Programs» National Instruments»LabVIEW



- Các thư viện *vi.lib* bên trong thư mục của LabVIEW
- Thay thế các thành phần bên trong *user.lib* hoặc *instr.lib* để các Vi được đưa vào bảng **Controls** và **Functions**

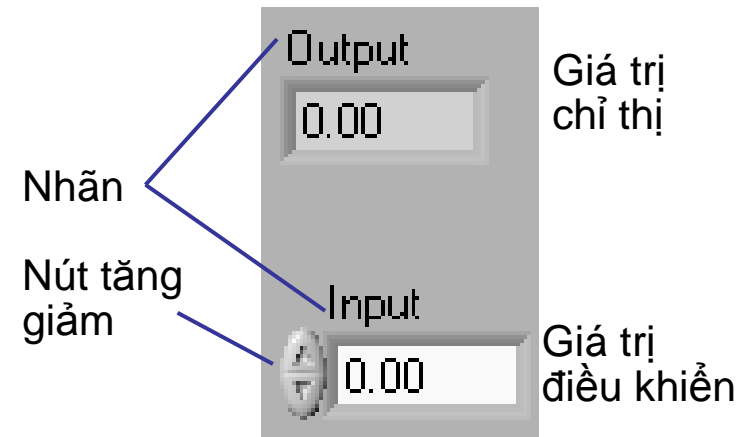
# Tạo một giao diện VI nhỏ nhỏ

Xây dựng một dự án với nút điều khiển controls (đưa vào) và chỉ thị indicators (đầu ra)



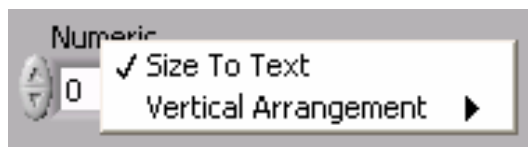
Điều khiển  
Boolean

Chỉ thị  
Boolean

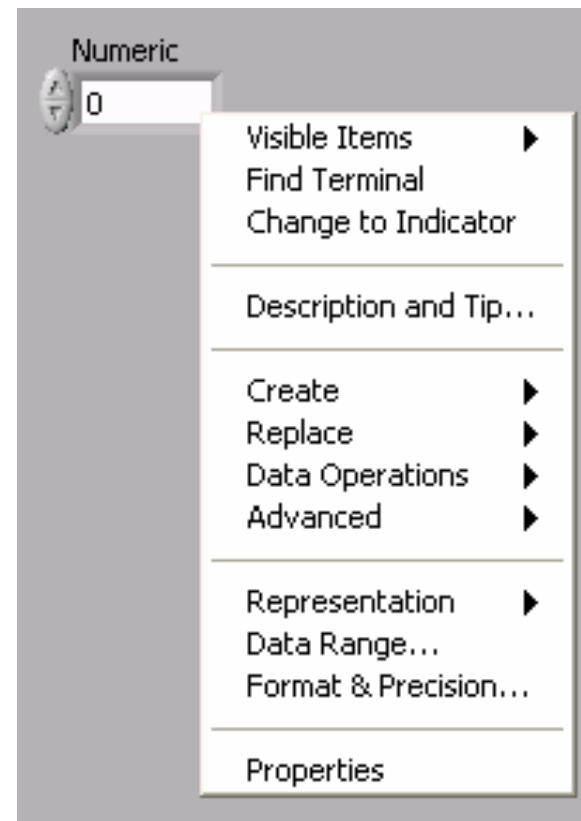


# Các Menu tắt cho đối tượng Front Panel

Click chuột phải vào  
nhãn để hiển thị menu  
tắt

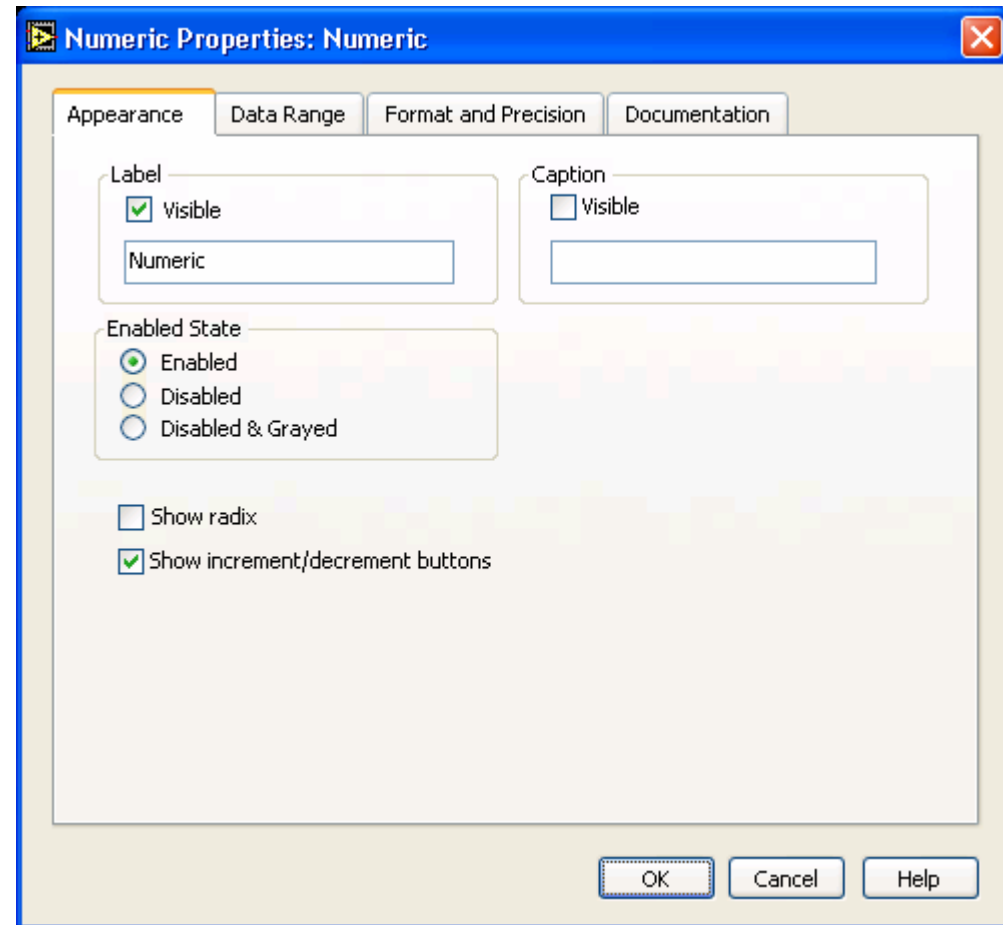


Click chuột phải vào khung  
hiển thị số để truy xuất menu  
tắt



# Trang đặc tính

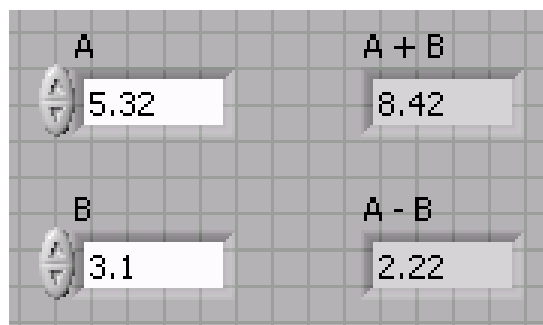
Click chuột phải lên một trình điều khiển hoặc chỉ thị trên Front panel và lựa chọn Properties từ menu tắt để xác lập hộp thoại đặc tính cho đối tượng.



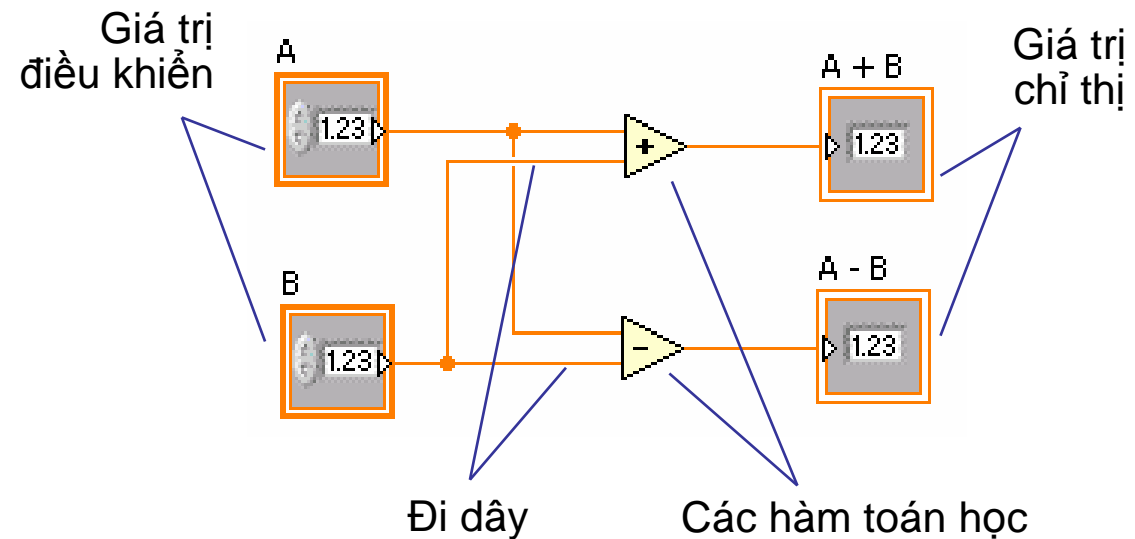


# Tạo một VI Block Diagram

## Front Panel

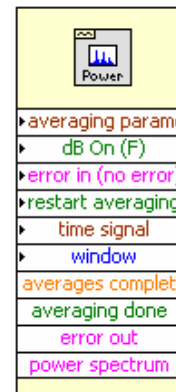
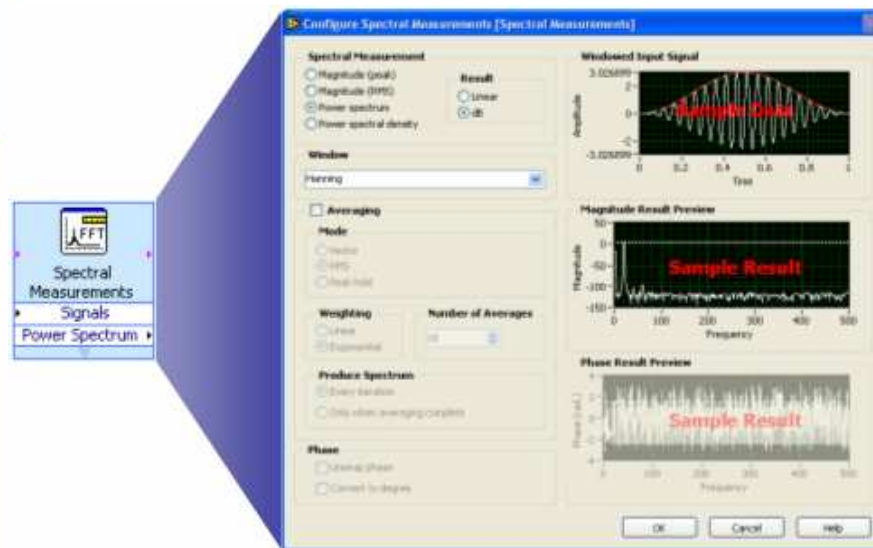


## Block Diagram

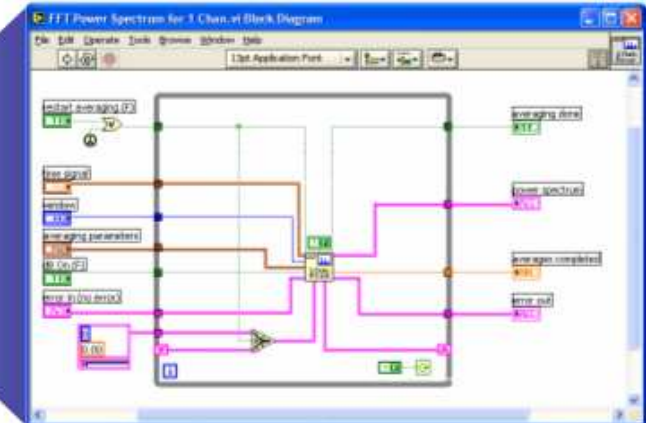


# Express VIs, VIs và Functions

- Express VIs: Các VI chuyên nghiệp tương tác với trang hộp thoại cấu hình
- Standard VIs: Các VI tiêu chuẩn được tùy biến đi dây
- Functions: Những thành phần hoạt động chủ yếu của LabVIEW (không phải Front panel hoặc Block diagram)



- ▶ averaging param
- ▶ dB On (F)
- ▶ error in (no error)
- ▶ restart averaging
- ▶ time signal
- ▶ window
- ▶ averages complet
- ▶ averaging done
- ▶ error out
- ▶ power spectrum

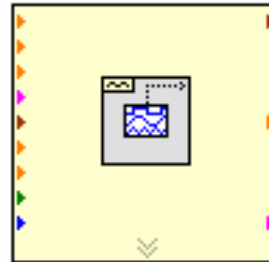


# Các điểm nút trong Block Diagram

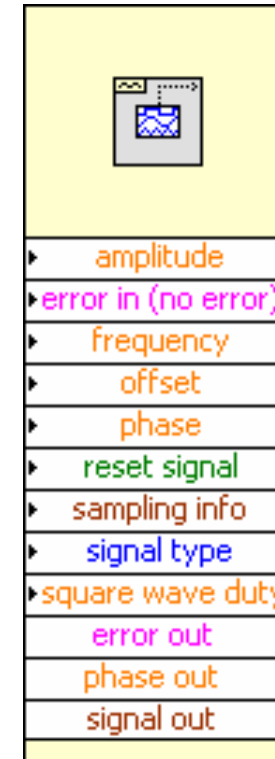
Icon



Nút mở rộng



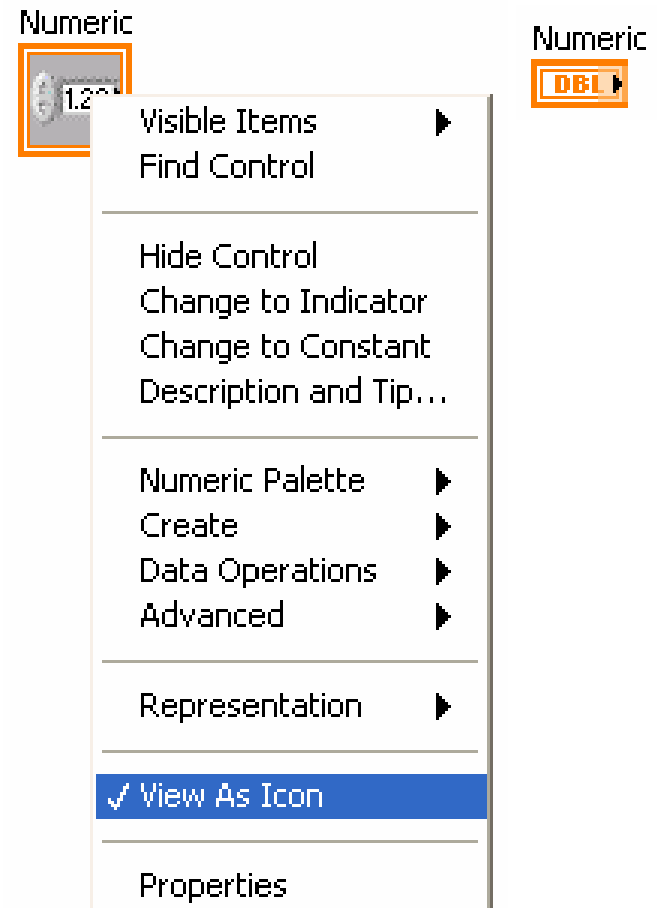
Nút mở rộng





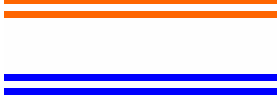









- Hàm VI cơ bản
- VI có 3 đường ra khác nhau
- Trường màu vàng chỉ định một VI tiêu chuẩn
- Trường màu xang chỉ định cho một VI Express

# Block Diagram Terminals

- Terminals là cổng ra và cổng vào để trao đổi thông tin giữa panel và diagram
- Terminals tương tự như tham số và hằng số trong ngôn ngữ lập trình cơ bản.
- Click chuột phải và chọn **View As Icon** để thay đổi tổng quan icon

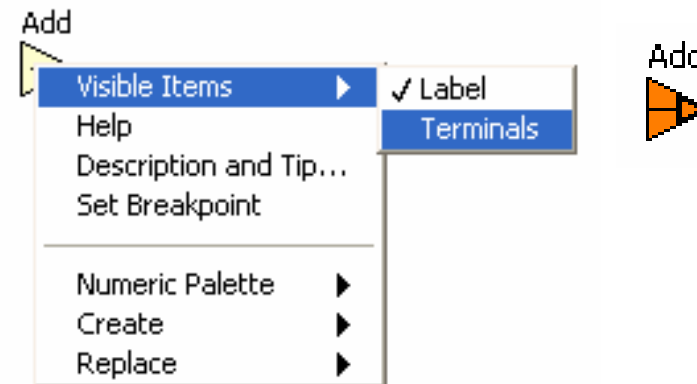
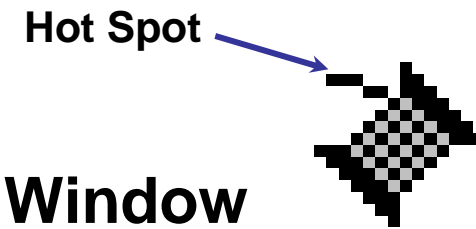


# Các dạng dây nối trên Block Diagram

	Vô hướng	Mảng 1 chiều	Mảng 2 chiều
Kiểu Numeric			
Kiểu Boolean			
Kiểu String			
Kiểu Dynamic			

# Kỹ thuật đi dây

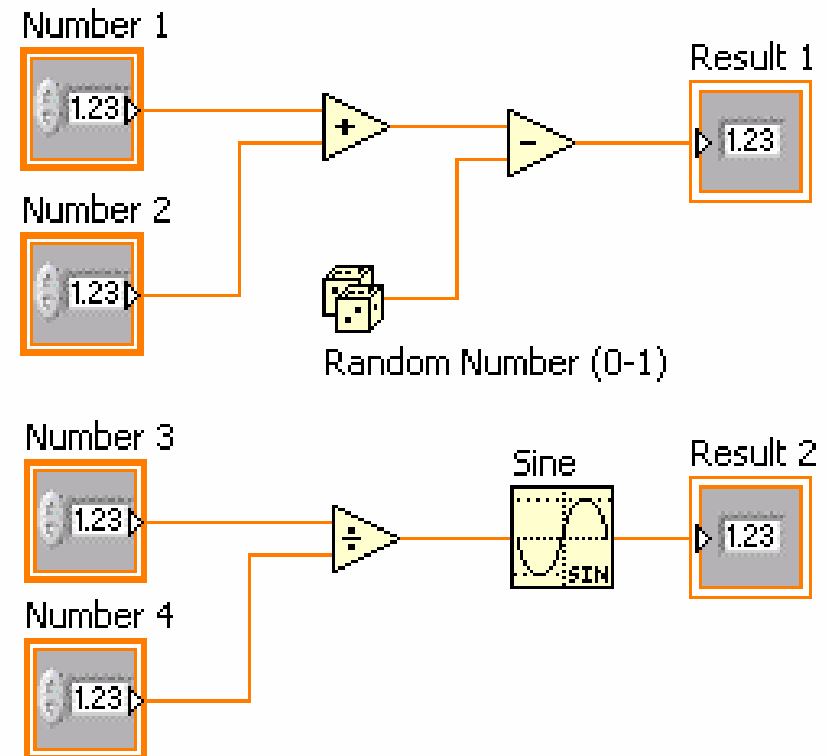
- Tự động đi dây
- Sử dụng cửa sổ trợ giúp **Context Help Window** khi đi dây
- Click chuột phải lên dây và chọn **Clean Up Wire (Ctrl+B)**
- Chú ý các dải Tip
- Tự động lộ trình đi dây
- Click chuột phải lên **Terminals** và lựa chọn **Visible Items»Terminals**



View the terminal connections to a function

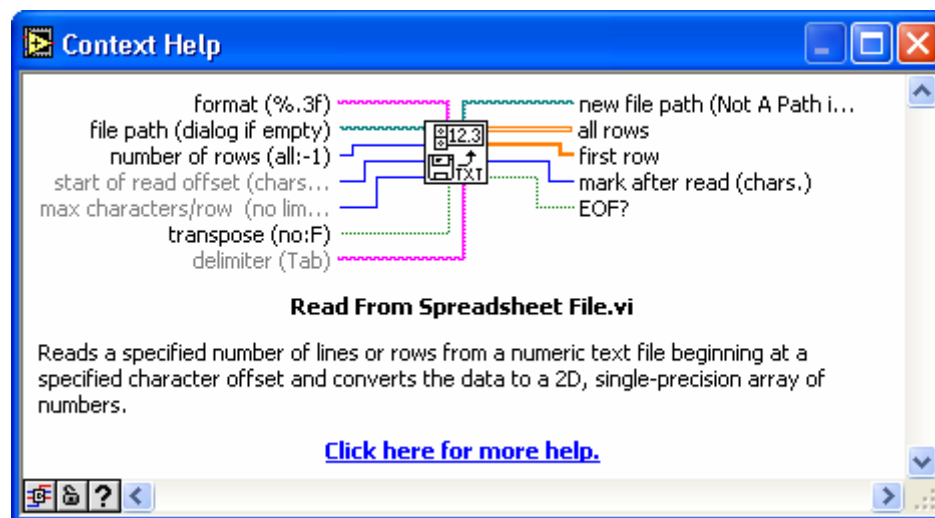
# Lập trình luồng dữ liệu

- Block diagram thực thi phụ thuộc vào luồng dữ liệu thì Block diagram sẽ **KHÔNG** thực thi từ trái qua phải
- Node sẽ thực hiện khi dữ liệu sẵn sàng đưa tới tất cả đầu vào của terminals.
- Node cung cấp dữ liệu cho tất cả đầu ra của terminals khi làm việc.



# Trợ giúp ngữ cảnh

- Để hiển thị cửa sổ trợ giúp ngữ cảnh ta lựa chọn **Help»Show Context Help**, hoặc ấn tổ hợp phím <Ctrl-H>, hoặc nhấp nút **Show Context Help Window** trên thanh công cụ
- Di chuyển con trỏ tới đối tượng để hiển thị trợ giúp
- Kết nối:  
**Cần thiết**– chữ béo  
**Đề nghị**– chữ thường  
**Tùy ý**- chữ mờ



Trợ giúp ngữ cảnh thông thường

Khoá trợ giúp

Trợ giúp chi tiết



# Trợ giúp trong LabView

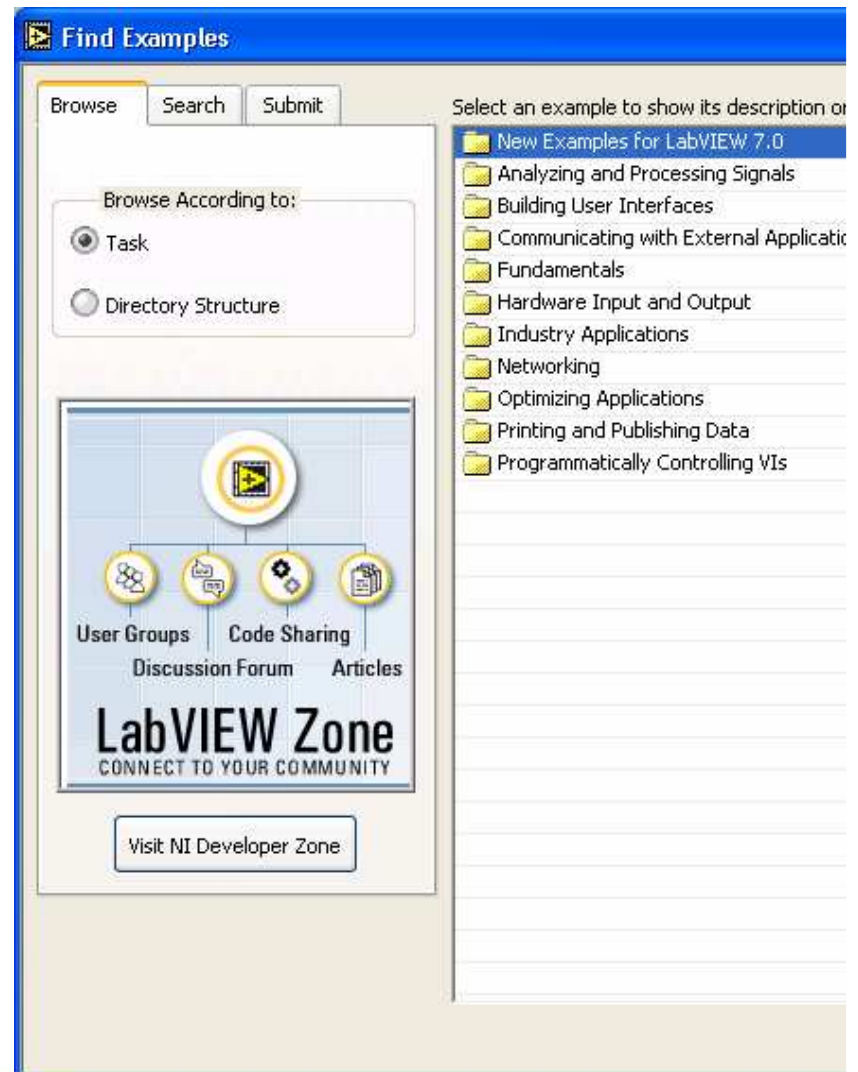
- Click vào trợ giúp chi tiết trên cửa sổ trợ giúp ngữ cảnh
- Lựa chọn **Help»VI, Function, & How-To Help**
- Click vào **Click here for more help** trong cửa sổ trợ giúp ngữ cảnh.

Tại đó mô tả chi tiết hầu hết các palettes, menus, tools, VIs, và functions, hướng dẫn từng bước các lệnh được sử dụng trong LabVIEW, các tính năng, các liên kết tới hướng dẫn, các văn bản PDF là sách hướng dẫn LabVIEW, những chú ý, và tài nguyên hỗ trợ về công nghệ.



# Tìm kiếm những ví dụ

- Để tìm kiếm những ví dụ có sẵn trong LabVIEW, ta lựa chọn **Help»Find Examples**
- Web-tích hợp
- Tìm theo từ khoá, kiểu ví dụ, kiểu phần cứng, v.v.



# Kỹ thuật gỡ rối

## Tìm lỗi



Nhấp nút có biểu tượng bên. Một cửa sổ sẽ xuất hiện hiển thị các bảng lỗi.

## Execution Highlighting



Click vào nút Execution Highlighting; Những giá trị hiển thị trên các dây nối.

# Kỹ thuật gỡ rối

## Probe – thăm dò



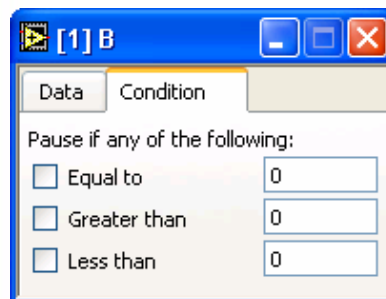
Click chuột phải lên dây và lựa chọn **probe**, nó thị dữ liệu của các đoạn nối dây

## Breakpoints – điểm gãy



Click chuột phải và lựa chọn Set Breakpoint; quá trình thi hành sẽ dừng lại ở điểm gãy.

## Conditional Probe – Thăm dò điều kiện



Đó là sự kết hợp giữa điểm gãy (breakpoint) và thăm dò (probe). Nhấp chuột phải lên dây và chọn **custom probe**.

# Kỹ thuật gỡ rối

Những nút Step Into, Over, and Out để thực hiện gỡ rối từng bước đơn một.



Click vào nút Step Into cho phép thực hiện từng bước đơn.



Click vào nút Step Over cho phép thực hiện từng bước đơn hoặc những điểm step over.



Click vào nút Step Out cho phép thực hiện từng điểm

# Tóm tắt

- Virtual instruments (VIs) có 3 phần chính - Front panel, Block diagram, biểu tượng và ô nối
- Front panel là giao diện người dùng của ngôn ngữ LabVIEW và Block diagram là mã thực thi.
- Block diagram bao gồm mã nguồn đồ họa của các nút, đầu cuối, và các đường đi dây
- Sử dụng Express VIs, những VIs chuẩn và các hàm ở trong Block diagram để tạo mã cho riêng bạn. Phần lớn nhu cầu chung đều sử dụng Express VIs sẵn có các hộp thoại cấu hình tương tác tạo ứng dụng cho riêng mình.
- Floating Palettes: Bảng công cụ, bảng điều khiển (chỉ có khi cửa sổ Front Panel đã active), và bảng hàm (chỉ có khi cửa sổ Block Diagram đã active)
- Tiện ích trợ giúp bao gồm cửa sổ trợ giúp ngữ cảnh và trợ giúp trong LabVIEW.



# Tóm tắt

- Đối tượng điều khiển (đầu vào) và đối tượng chỉ thị (đầu ra) ở trong cửa sổ Front panel.
- Sử dụng Operating tool để vận dụng các đối tượng Panel. Sử dụng Positioning tool để lựa chọn, di chuyển, và định lại cỡ của đối tượng panel. Sử dụng Wiring tool để kết nối các đối tượng trong đồ thị.
- Các đối tượng điều khiển đầu cuối cho đường viền dày hơn đối tượng chỉ thị đầu cuối.
- Tất cả các đối tượng Front panel objects đều có trang thuộc tính và các menu tắt.
- Đi dây là một kỹ thuật cho điều khiển luồng dữ liệu và đưa ra kết quả trong ngôn ngữ LabVIEW.
- Khi mà nút mũi tên Run có biểu tượng bị gãy điều đó có nghĩa là ứng dụng VI không thể thực thi
- Những tùy chọn và công cụ gỡ rối khác nhau cho giá trị thiết lập như nhau: thăm dò (probe) và điểm gãy (breakpoint), đèn báo thực thi, và các bước gỡ rối đơn.



# Mẹo vặt

- Các phím tắt

<i><u>Windows</u></i>	<i><u>Sun</u></i>	<i><u>Linux</u></i>	<i><u>MacOS</u></i>	
<Ctrl-R>	<"-R>	<M-R>	<z-R>	Chạy file VI
<Ctrl-F>	<"-F>	<M-F>	<z-F>	Tìm đối tượng
<Ctrl-H>	<"-H>	<M-H>	<z-H>	Mở cửa sổ trợ giúp ngữ cảnh
<Ctrl-B>	<"-B>	<M-B>	<z-B>	Loại bỏ các đường đi dây lỗi
<Ctrl-W>	<"-W>	<M-W>	<z-W>	Đóng cửa sổ làm việc
<Ctrl-E>	<"-E>	<M-E>	<z-E>	Chuyển đổi cửa sổ Giao diện/Mã

- Gọi bảng công cụ bằng cách giữ <shift>-click chuột phải
- Tăng/Giảm nhanh sử dụng khóa <shift>
- Chọn Tools»Options — để thiết lập tính ưu tiên trong LabVIEW
- File»VI Properties – thuộc tính VI





# Bài học 2

## Lập trình theo Modul

### Các chủ đề liên quan

SubVIs

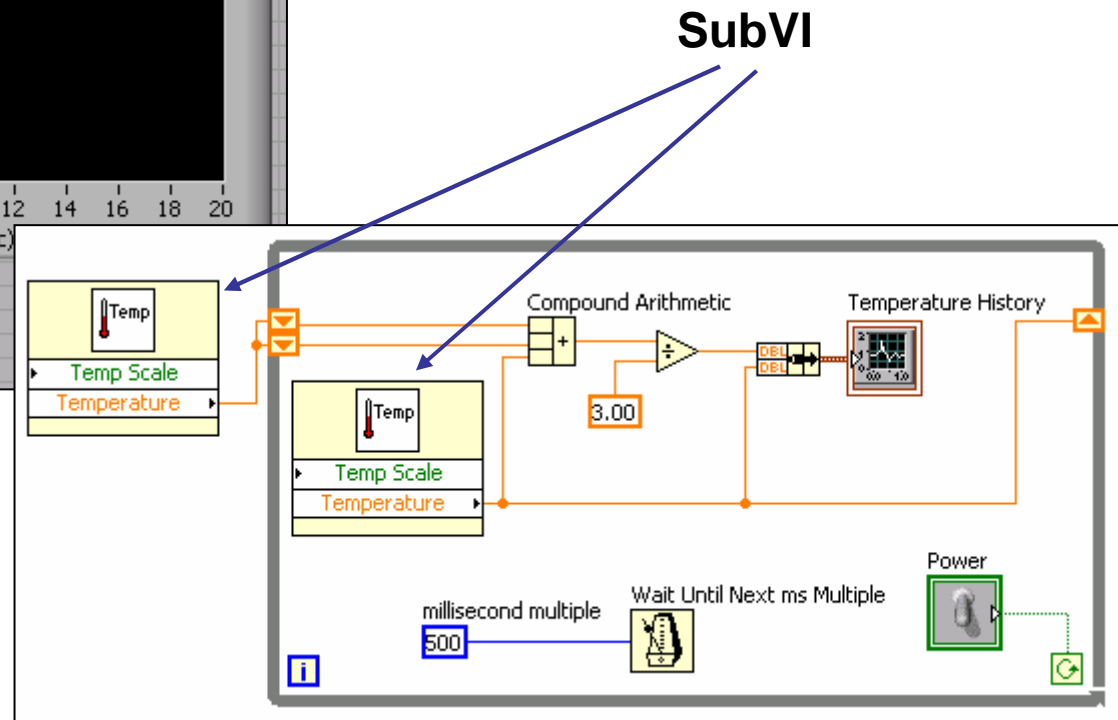
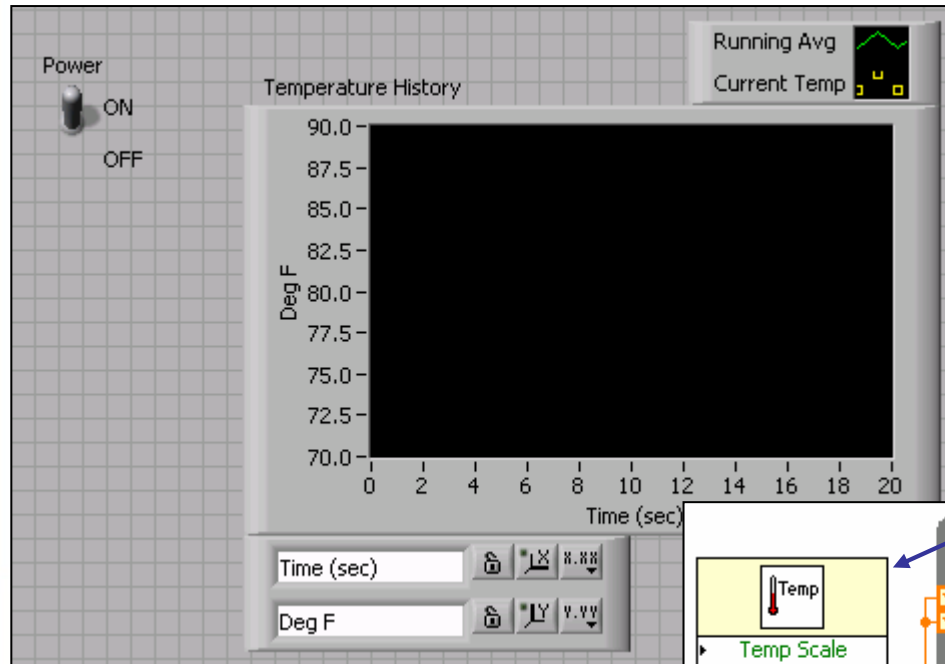
Icon và Connector Pane

Sử dụng SubVIs

Tạo một SubVI từ các phần trong một VI



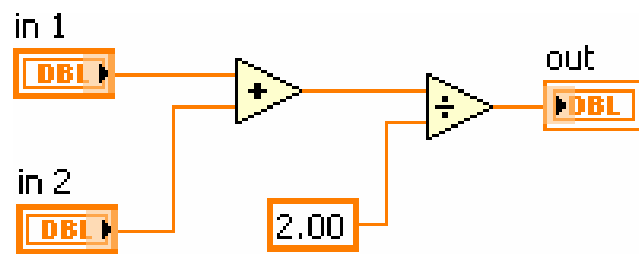
# Thứ bậc trong LabVIEW



# SubVIs

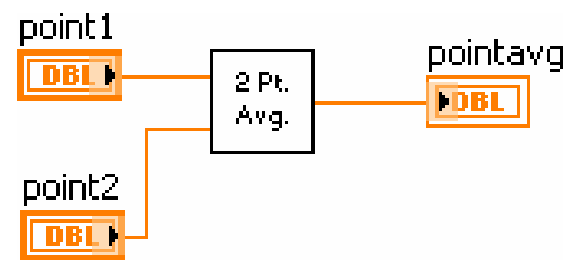
**Function Pseudo Code**  
function average (in1, in2, out)  
{  
out = (in1 + in2)/2.0;  
}

**SubVI Block Diagram**

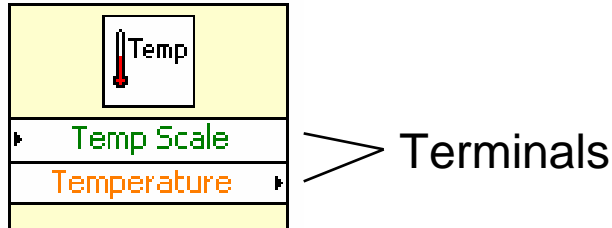


**Calling Program Pseudo Code**  
main  
{  
average (point1, point2, pointavg)  
}

**Calling VI Block Diagram**



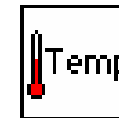
# Icon/Connector



Một icon miêu tả gợi nhớ cho VI đó trong mỗi block diagram khác nhau

Một connector đưa dữ liệu tới và nhận dữ liệu từ một subVI thông qua vùng đưa ra

Icon



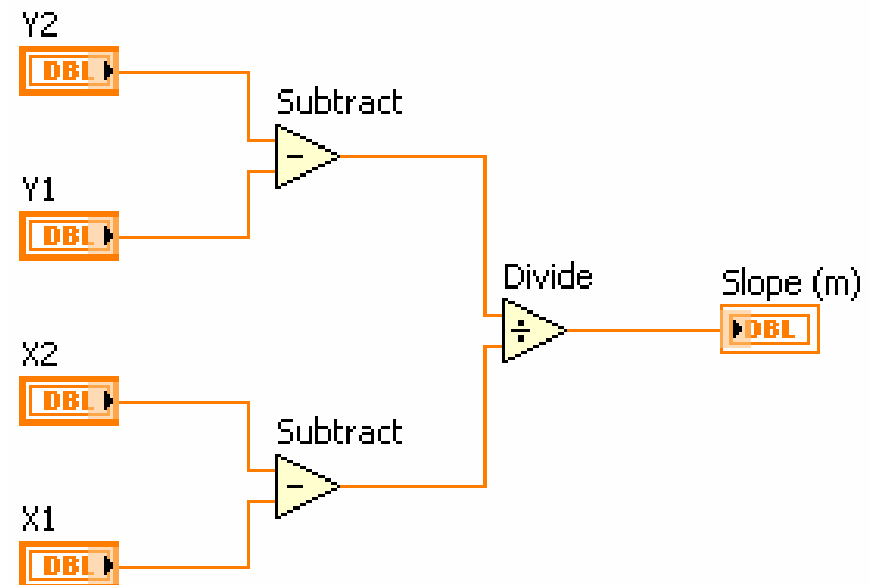
Connector



Terminals

# Ví dụ một SubVI – một phép tính cơ bản

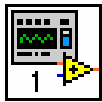
- Một VI nằm trong một VI khác được gọi là subVI
- Để sử dụng một VI giống như một subVI, ta tạo một icon và connector pane sau đó xây dựng Front panel và Block diagram



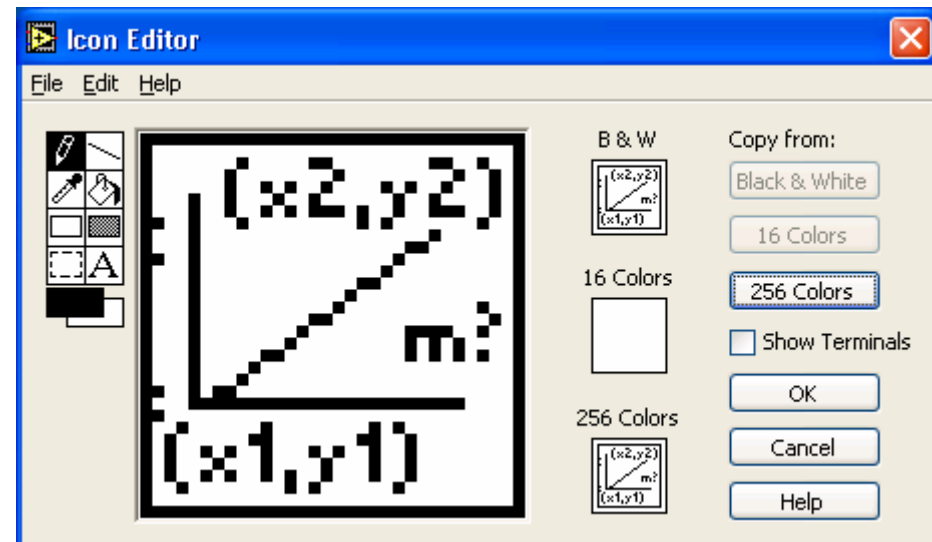
# Tạo Icon

- Icon: biểu tượng miêu tả gợi nhớ của 1 VI
- Click chuột phải lên icon pane (Panel hoặc Diagram) ở góc cửa sổ
- Nên tạo một icon đen trắng

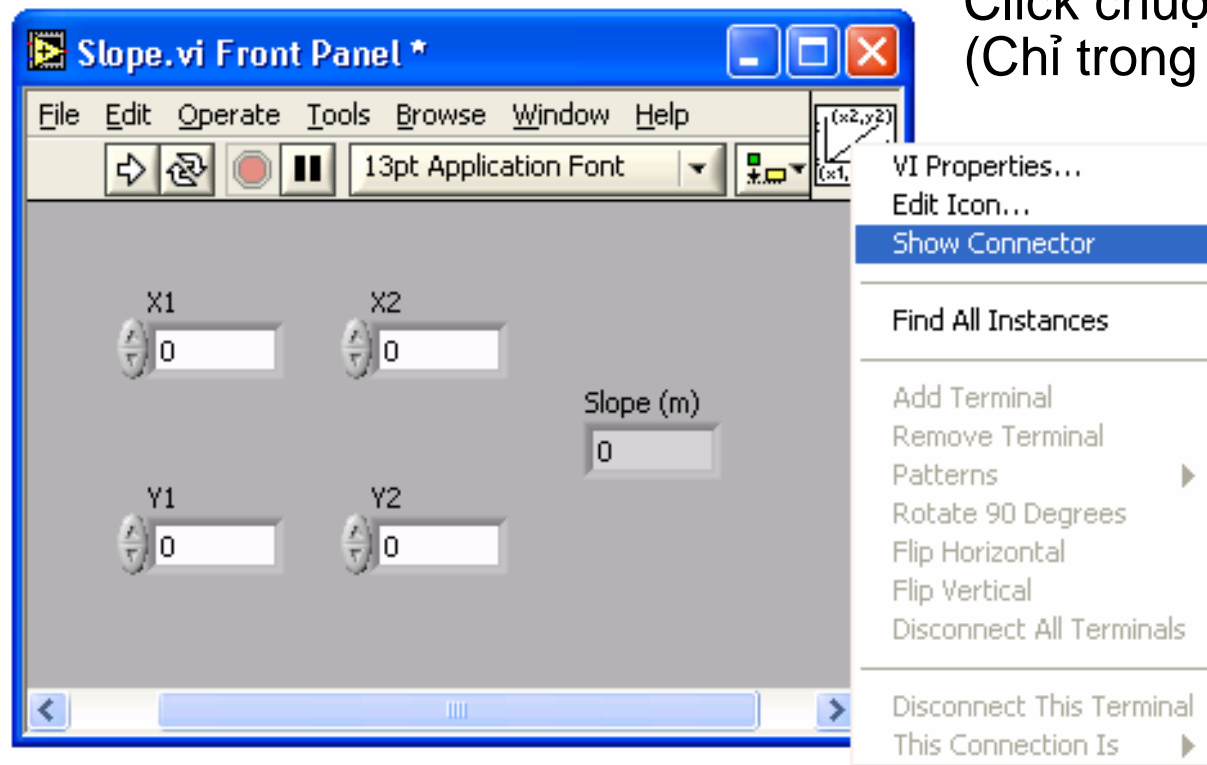
Icon mặc định



Tạo một icon theo người làm



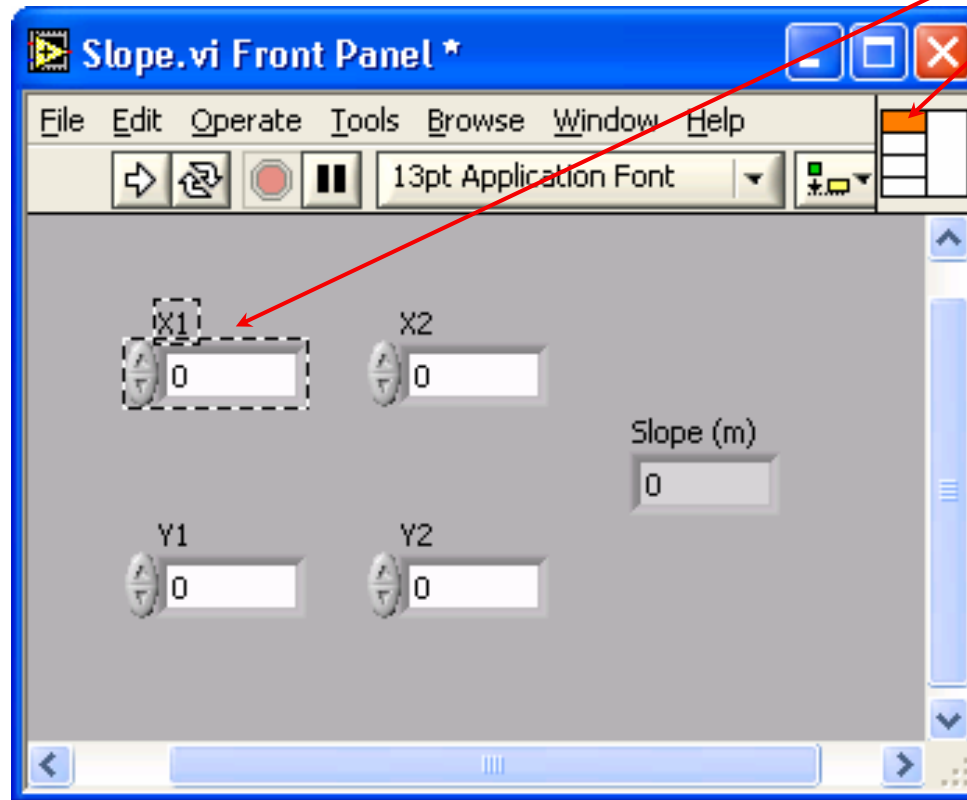
# Tạo Connector



Click chuột phải lên icon  
(Chỉ trong Front Panel)

# Tạo Connector – tiếp...

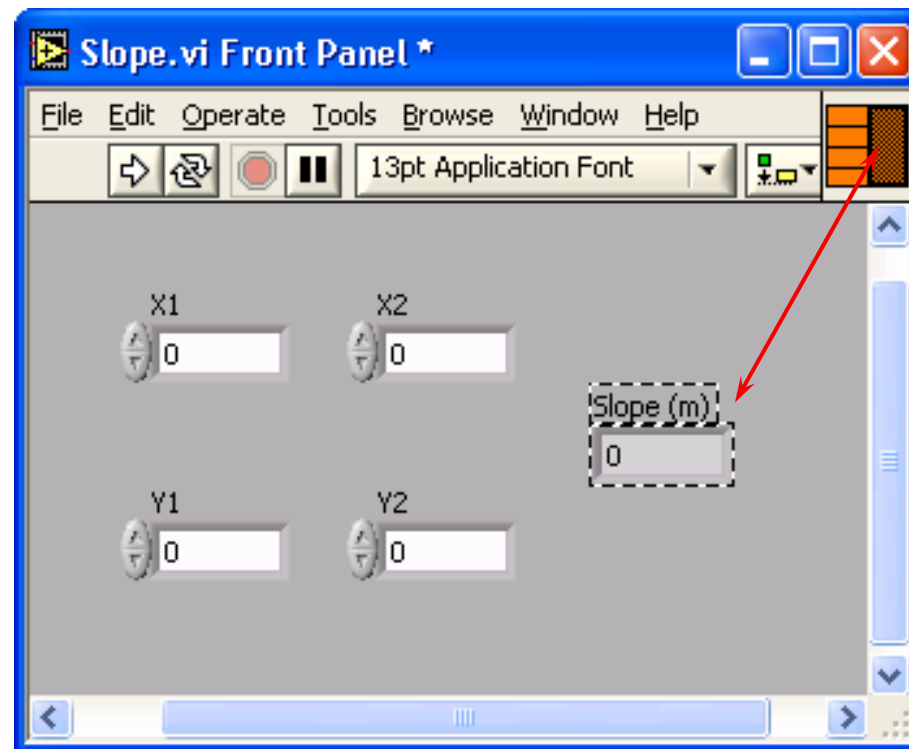
Chọn điểm





# Tạo Connector – tiếp...

Đầu ra có màu khớp với kiểu dữ liệu mà chúng liên kết tới.  
Click vào đầu ra để nhìn thấy đối tượng Front panel kết hợp



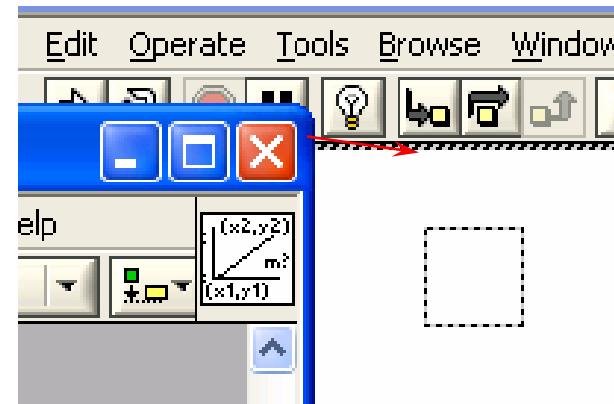
# Sử dụng một VI giống như SubVI

All Functions » Select a VI...

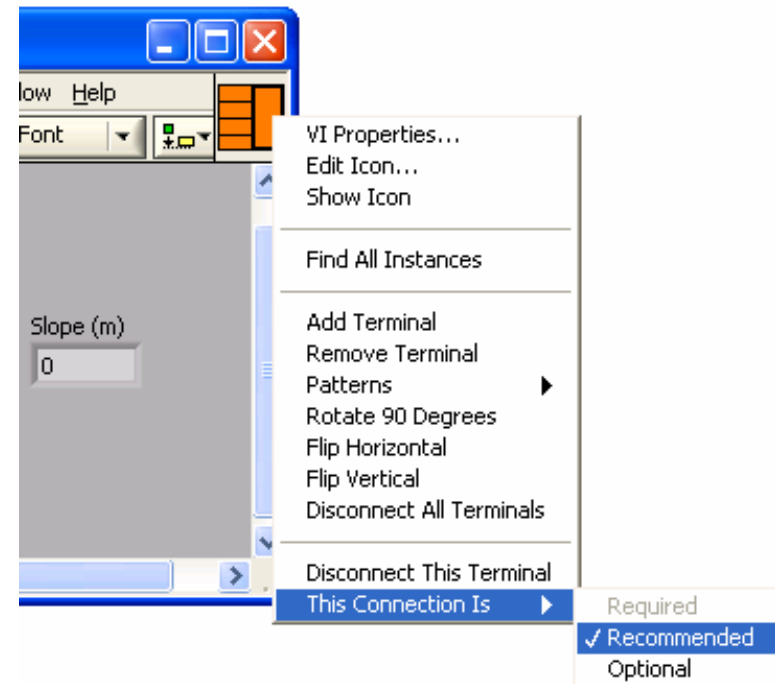
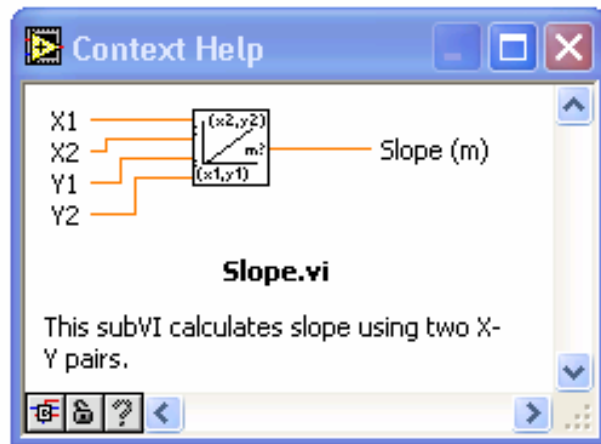


<Hoặc>

Kéo thả **icon** tới Block diagram đích



# Trợ giúp và phân loại đầu cuối

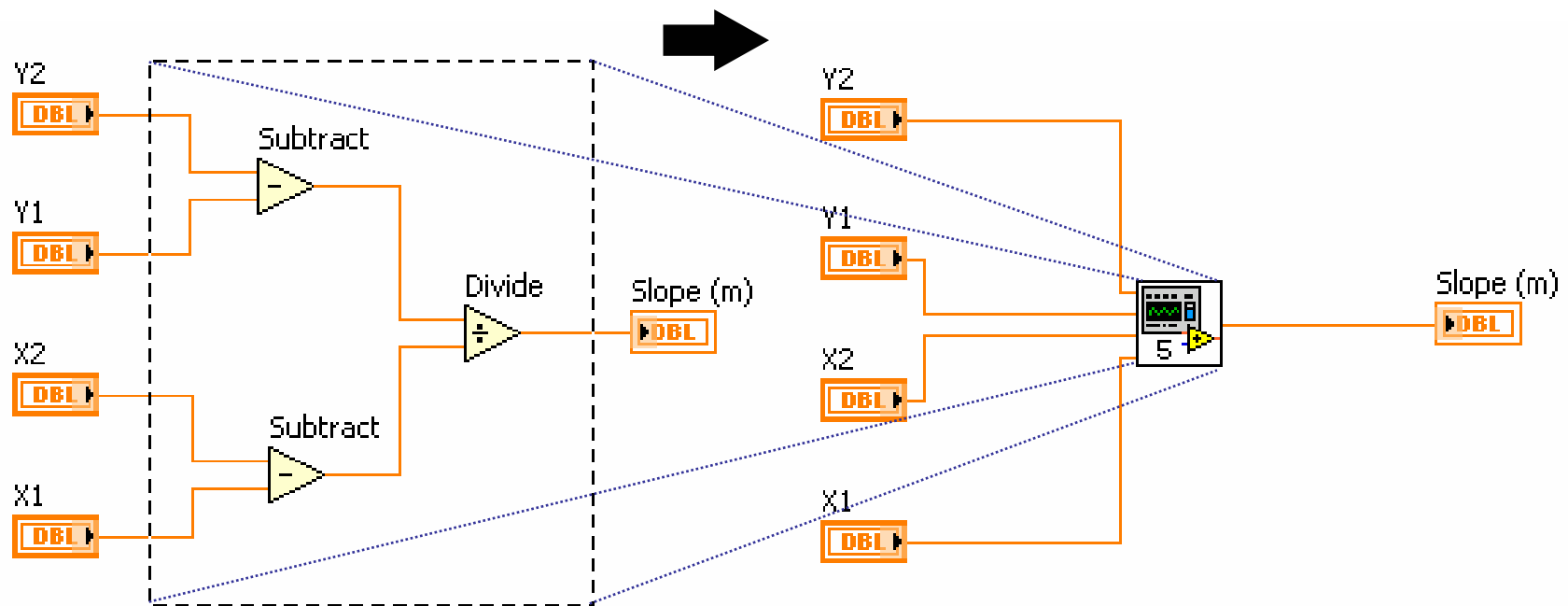


Phân loại đầu vào/ra:

- Required — Lỗi nếu không kết nối
- Recommended — Cảnh báo nếu không kết nối
- Optional — Không có tác dụng nếu không kết nối

# Tạo SubVI tùy chọn

- Khoanh vùng muốn tạo một subVI
- Chọn **Edit** » **Create SubVI** để thấy kết quả



# Tóm tắt

- Các VI có thể được sử dụng giống như những subVI sau khi bạn đã tạo icon và connector
- Để tạo Icon sử dụng Icon Editor
- Connector được định bởi số lựa chọn của đầu cuối
- Nạp vào subVI sử dụng tùy chọn **Select a VI** ở trong bảng **All Functions** hoặc kéo thả **icon** vào một diagram mới.
- Trợ giúp trực tuyến cho subVI sử dụng lựa chọn Show Context Help
- Mô tả chức năng đối tượng
- Sử dụng tính năng **Create SubVI** để dễ dàng tạo những Modul của Block diagram



# Bài học 3

## Lặp lại và các vòng lặp

### CÁC CHỦ ĐỀ

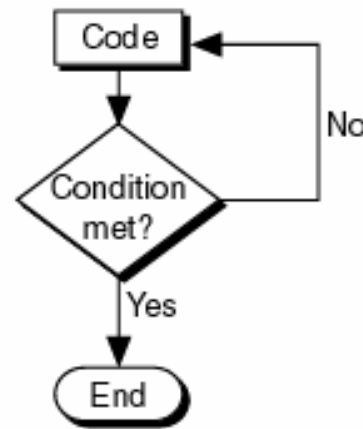
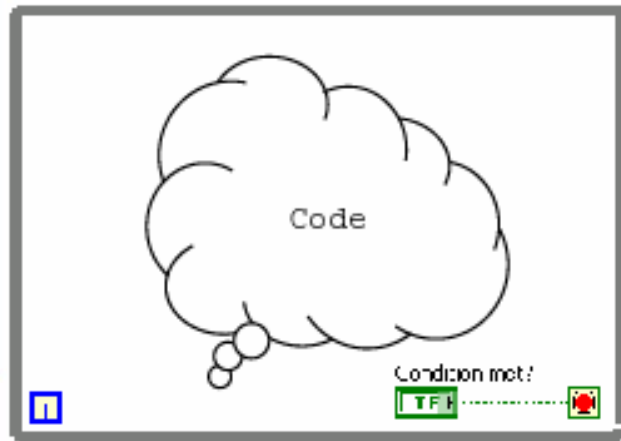
Vòng lặp While

Vòng lặp For

Truy xuất dữ liệu vòng lặp trước đó



# Vòng lặp While



**Repeat (code);  
Until Condition met;  
End;**

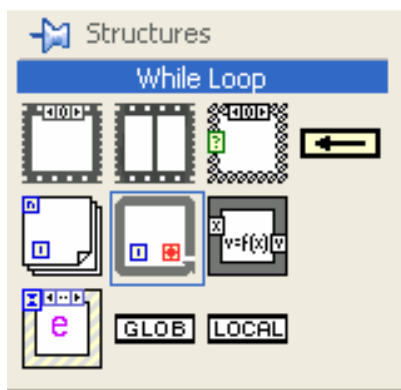
Vòng lặp While trong LabVIEW

Lưu đồ thuật toán

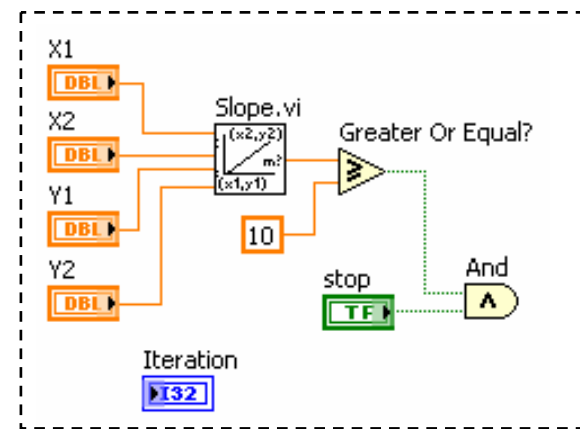
Sinh mã

# Vòng lặp While

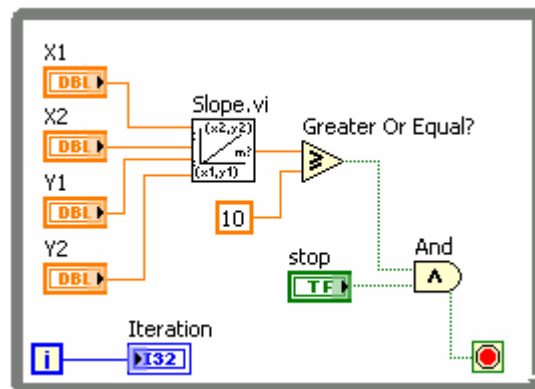
1. Chọn While Loop



2. Khoanh vùng mã muốn đặt trong vòng lặp



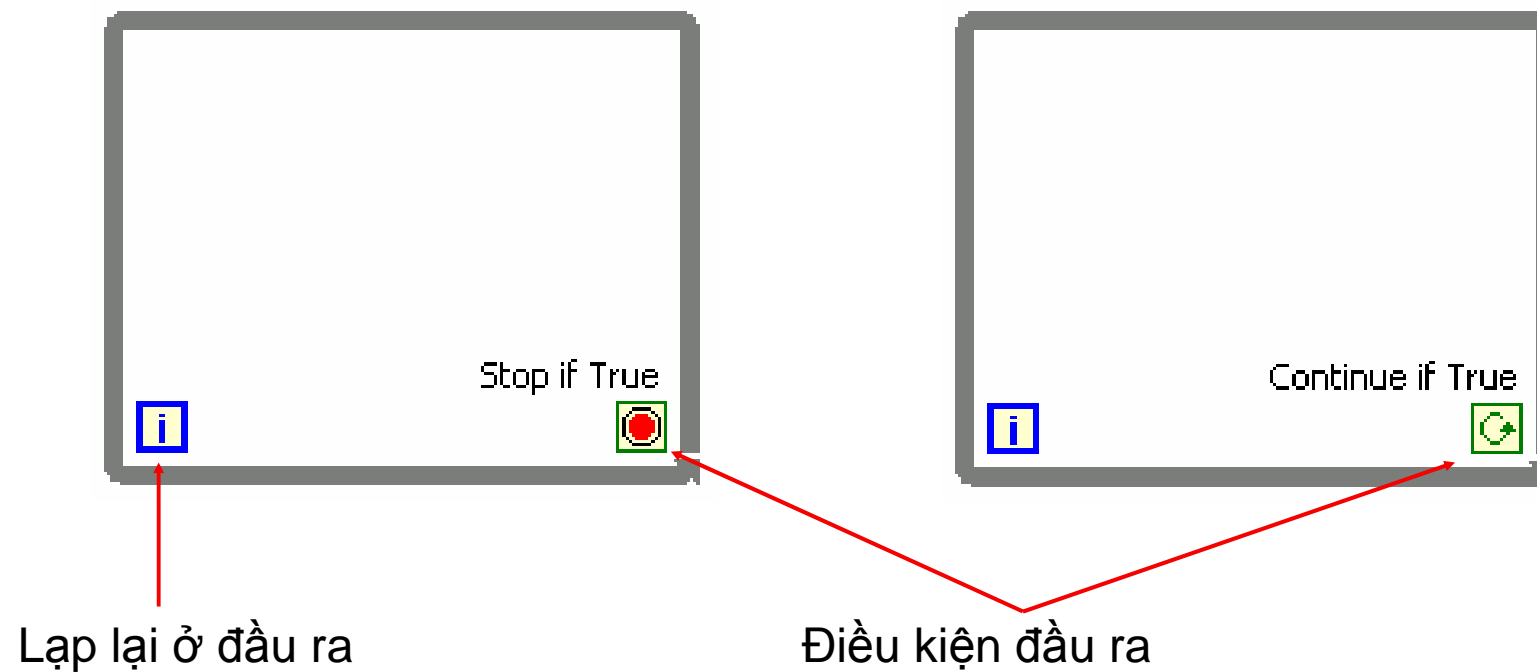
3. Kéo thả vào vòng lặp các nút và sau đó đi dây





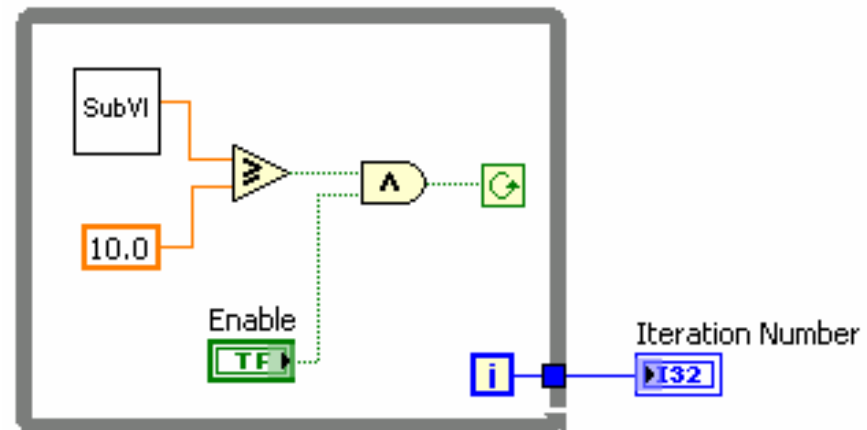
# Lựa chọn vòng lặp điều kiện

Mặc định: Stop nếu là True

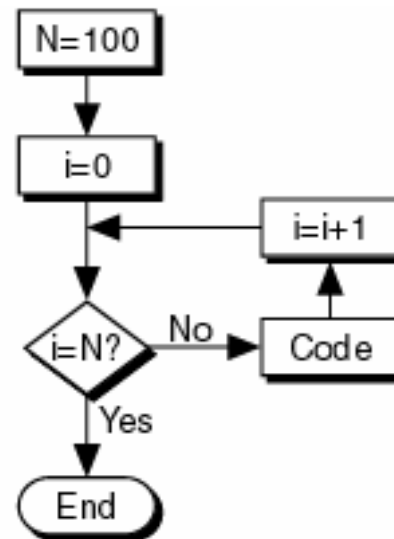
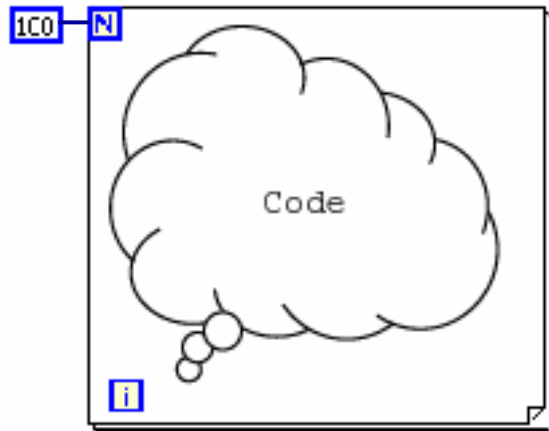


# Cấu trúc Tunnel

- Các Tunnel cho dữ liệu đi vào và đi ra khỏi cấu trúc.
- Tunnel là một khối được xuất hiện trên khung; màu của khối có liên quan tới kiểu dữ liệu được nối tới tunnel.
- Khi mà một tunnel đưa dữ liệu vào vòng lặp, thì vòng lặp chỉ thực hiện sau khi dữ liệu đến.
- Dữ liệu đi ra ngoài của vòng lặp sau khi kết thúc vòng lặp.



# Vòng lặp For

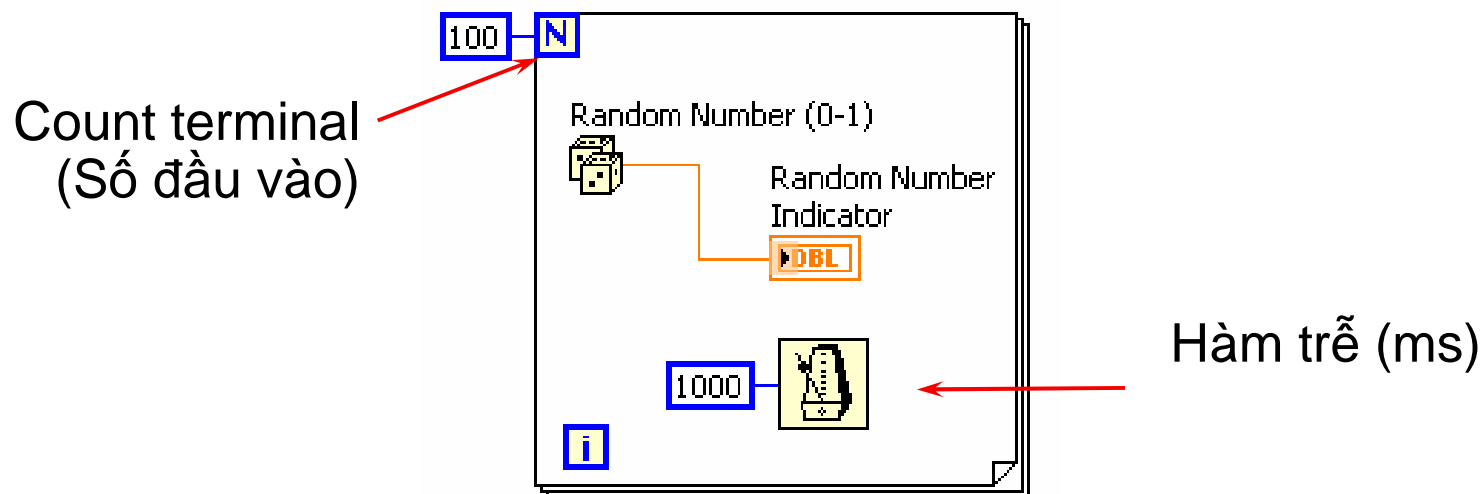


```
N=100;  
i=0;  
Until i=N:  
  Repeat (code; i=i+1);  
End;
```

Vòng lặp For trong LabVIEW      Lưu đồ thuật toán      Sinh mã

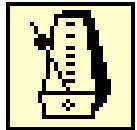
# Vòng lặp For

- Nằm trong Structures của Functions
- Mã được tạo trong vòng lặp
- Quá trình lặp phụ thuộc vào thời gian

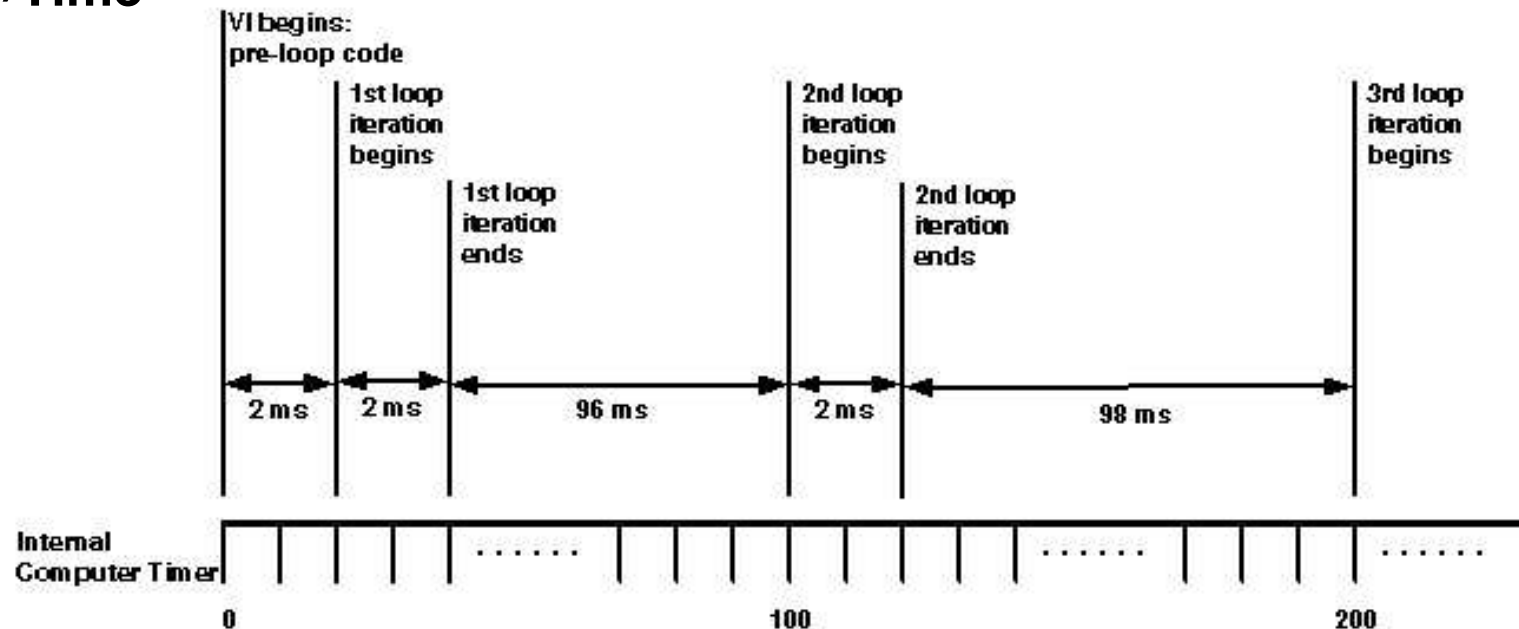
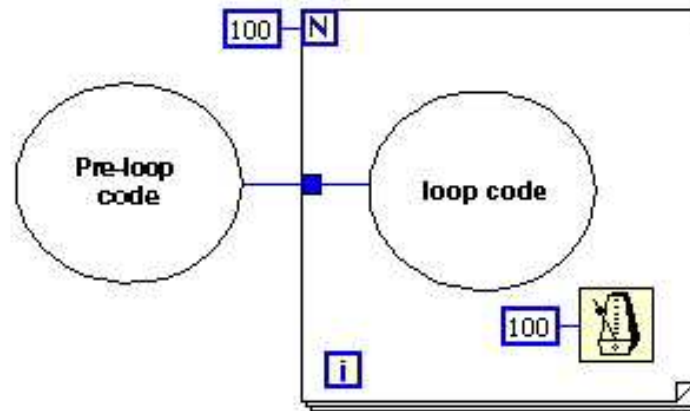


# Hàm trễ

Wait Until Next  
ms Multiple

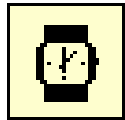


Functions»Time  
& Dialog



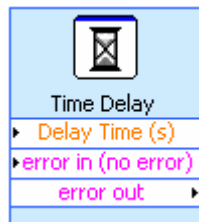
# Hàm trễ

Wait (ms)

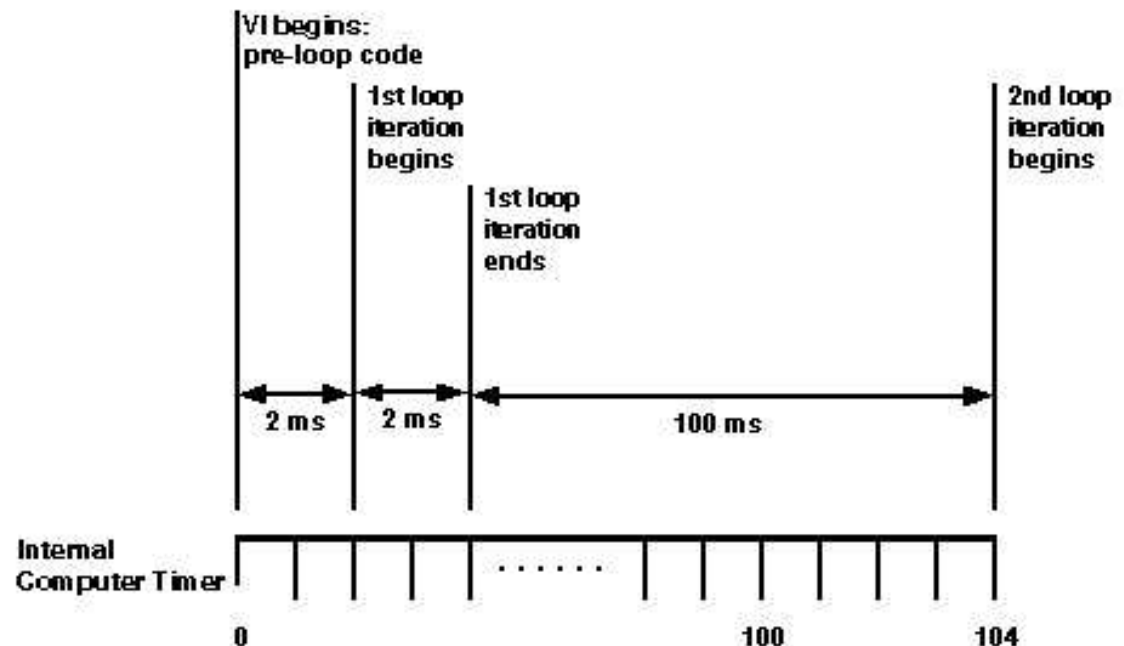
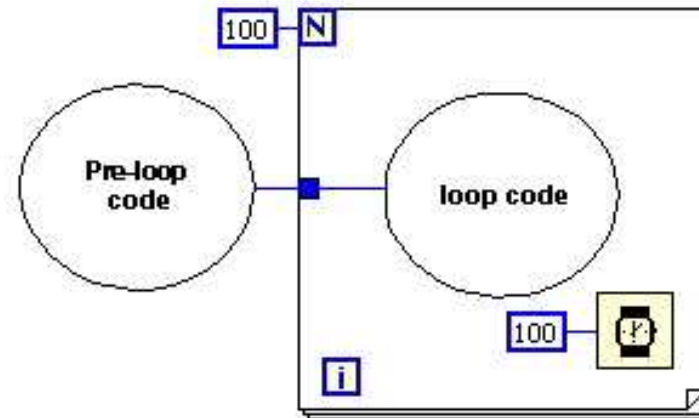


Functions»Time & Dialog palette

Time Delay



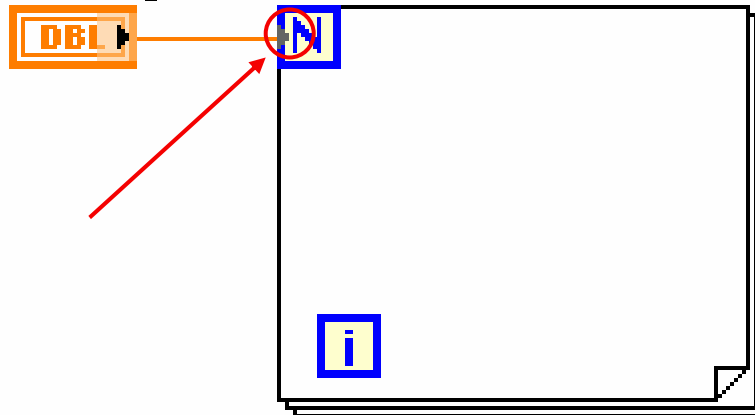
Functions»Time & Dialog palette



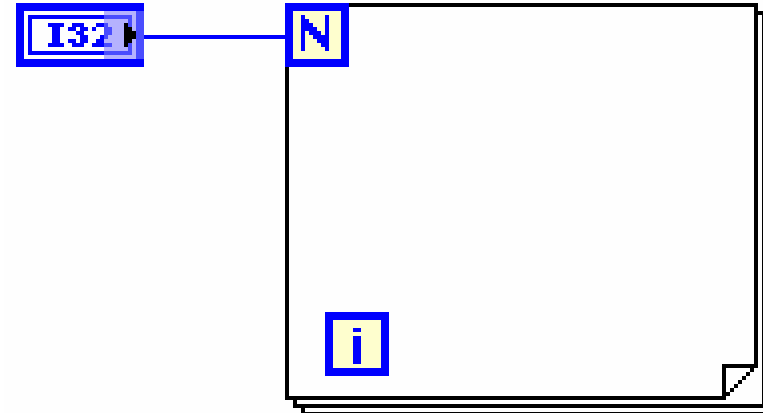
# Chuyển đổi số

- Các số mặc định thường double (8 bytes) hoặc long integer (4 bytes)
- LabVIEW chuyển đổi tự động các giá trị tương đương
- Vòng lặp For tính đầu cuối luôn luôn chuyển thành dạng long integer

Double-Precision,  
Floating-Point Numeric

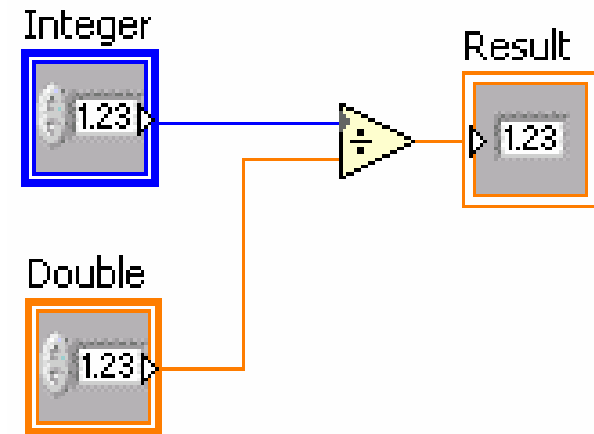


Long Integer



# Chuyển đổi số

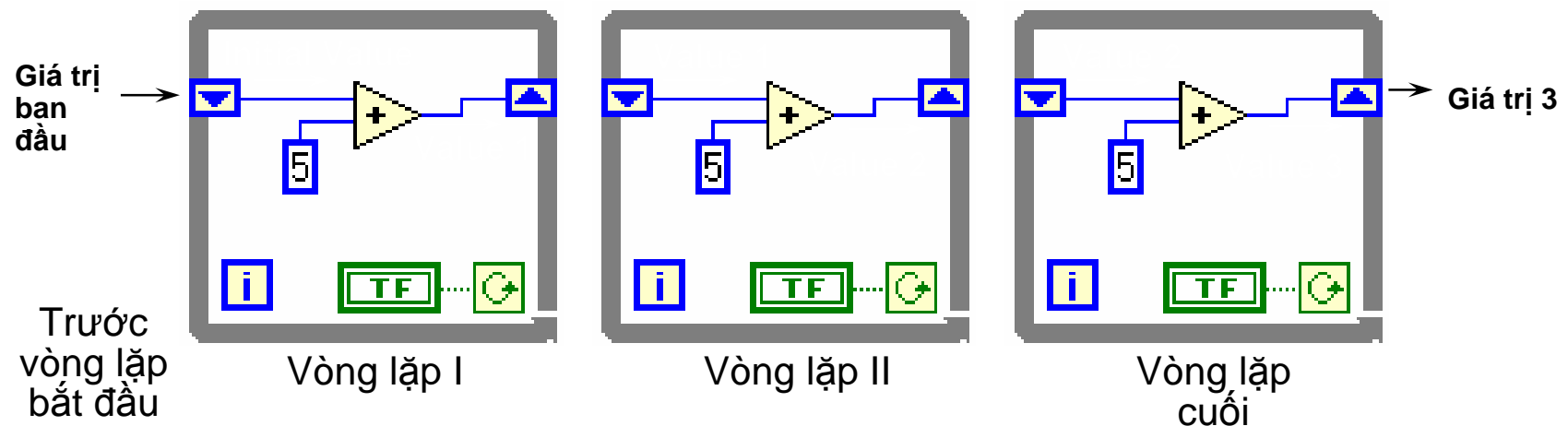
- LabVIEW lựa chọn tình trạng cụ thể để sử dụng lượng bit.
- Nếu như lượng bit là tương đương thì LabVIEW sẽ lựa chọn phần không đánh dấu thay vì phần đánh dấu.
- Để lựa chọn phần thể hiện, click chuột phải lên đối tượng cuối và lựa chọn mục **Representation**.
- Khi mà LabVIEW chuyển đổi những điểm thay đổi từ numeric thành integer, nó sẽ làm chẵn về kiểu integer gần nhất. LabVIEW làm chẵn tới x.5 dạng integer gần nhất.  
Ví dụ như, LabVIEW đưa 2.5 thành 2 và 3.5 thành 4.





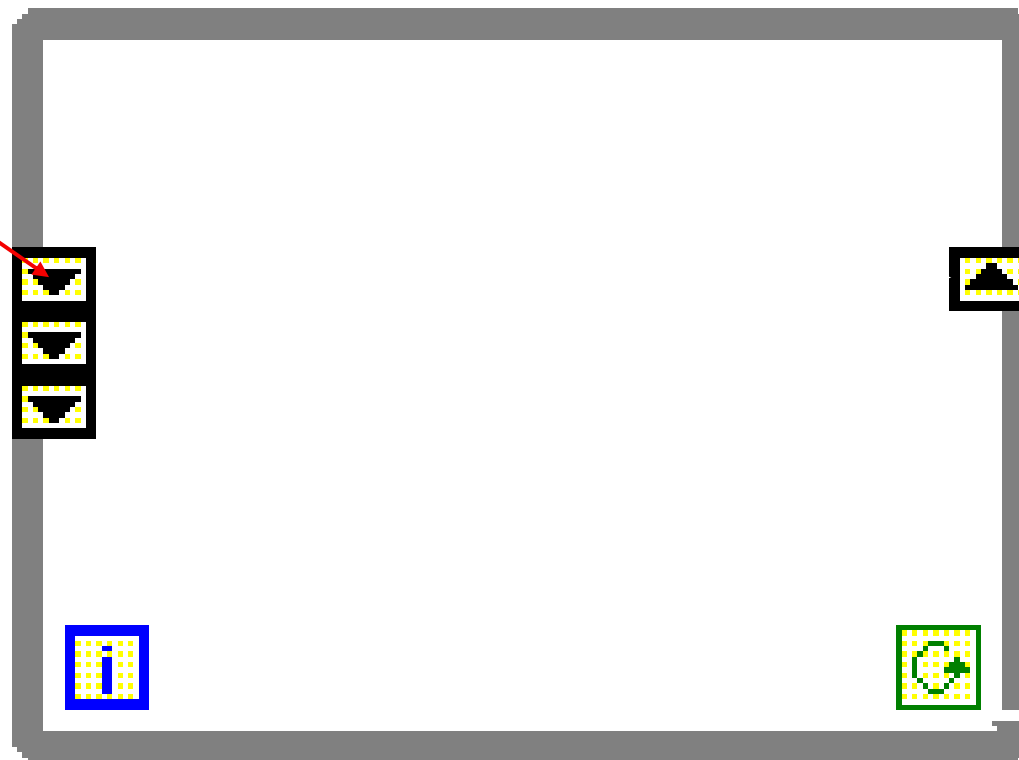
## Truy xuất dữ liệu vòng lặp trước đó – Thanh dịch chuyển

- Nó có giá trị ở bên phải hoặc trái của khung cấu trúc vòng lặp
- Click chuột phải vào khung và chọn **Add Shift Register**
- Đầu ra bên phải sẽ lưu trữ dữ liệu trong sự hoàn thành của phép lặp.
- Đầu ra bên trái sẽ cung cấp dữ liệu đã được lưu trữ ở dạng ban đầu của bước lặp kế tiếp



# Thêm vào thành phần của Shift Register

Click chuột phải lên vòng lặp ở bên trái khung để tham vào đối tượng



Click chuột phải lên phía bên phải khung để thêm vào một shift register

# Nút phản hồi

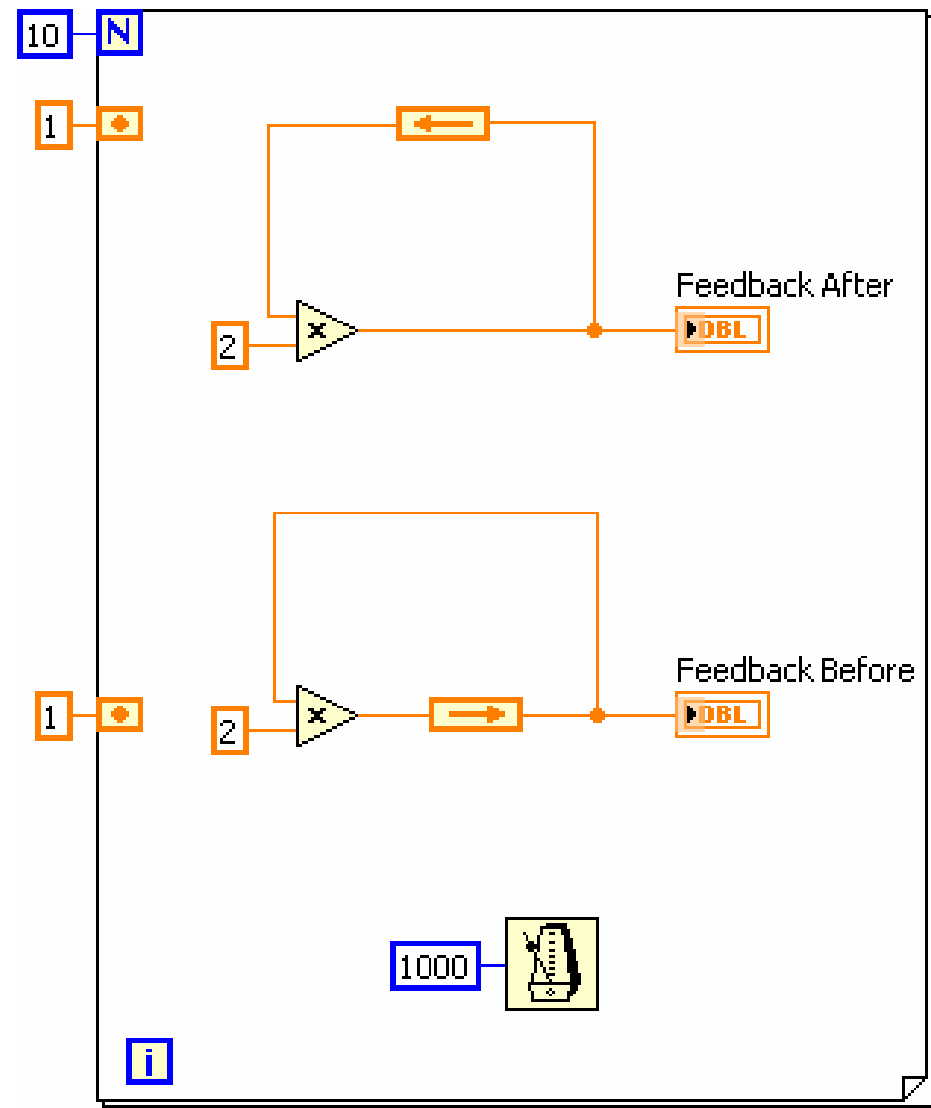


- Sẽ xuất hiện tự động trong vòng lặp For hoặc While nếu như bạn đi dây ở đầu ra của một subVI, function, hoặc group của subVI và functions tới đầu vào của VI tương đương, function, hoặc group.
- Dữ liệu lưu trữ khi mà hoàn tất một vòng lặp, gửi dữ liệu tới vòng lặp kế tiếp và truyền đi mọi kiểu dữ liệu.

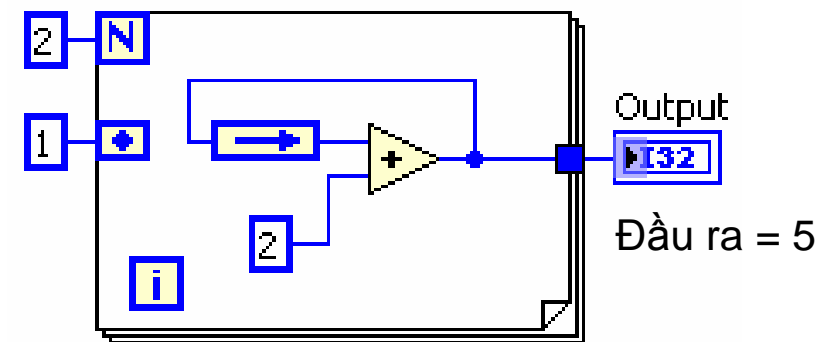
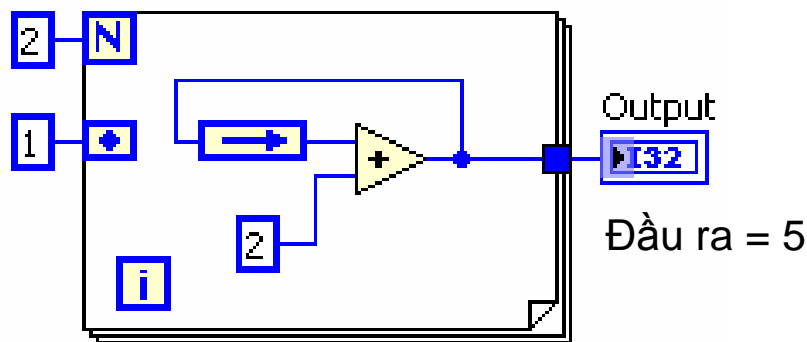
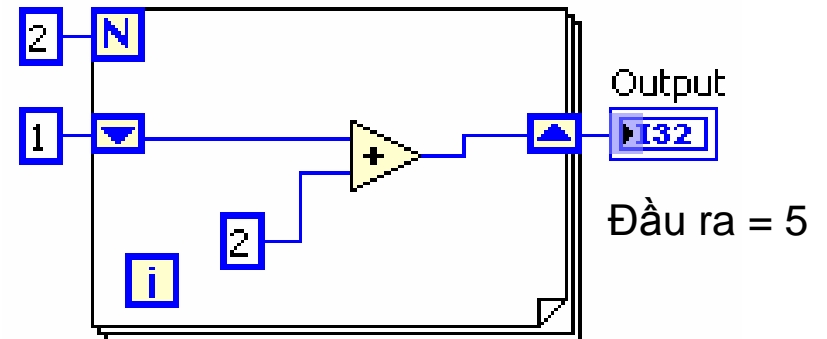
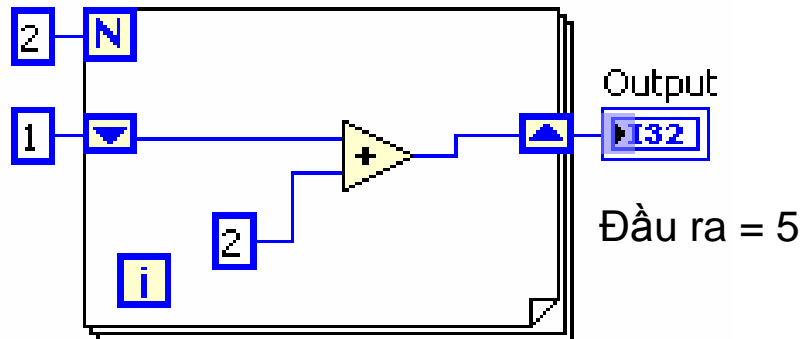


# Nút phản hồi

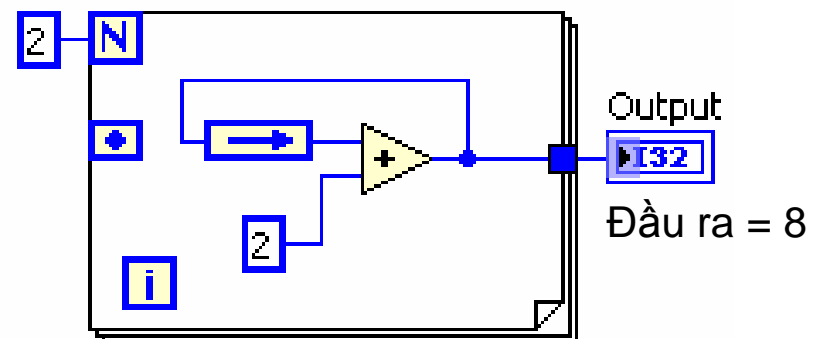
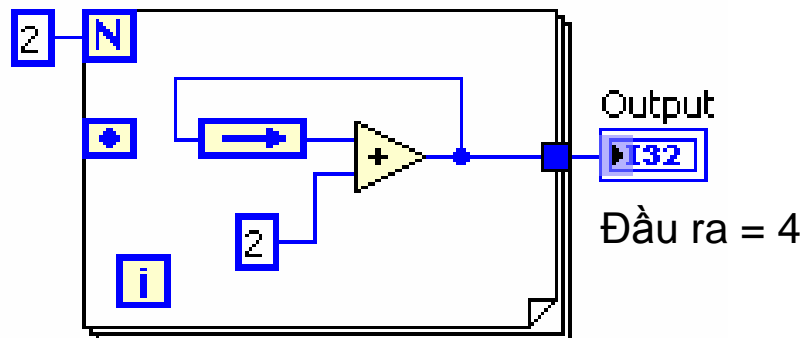
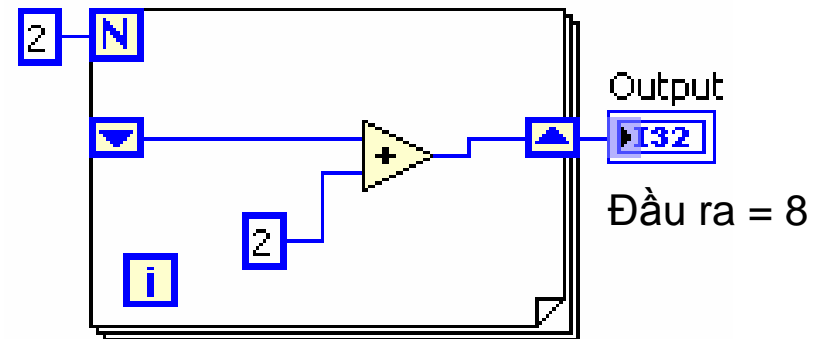
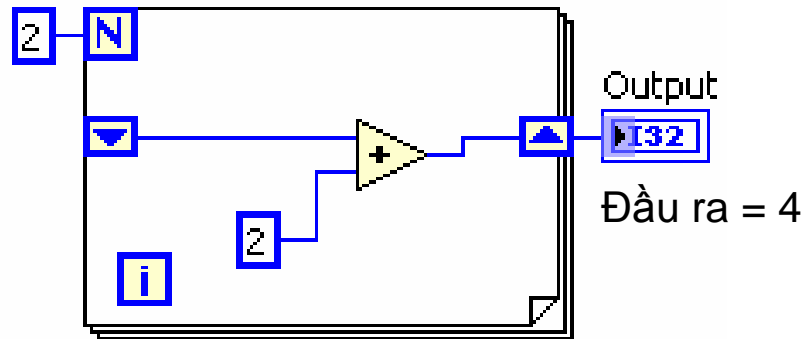
- Đi dây từ đầu ra tới đầu vào được tự động tạo ra một nút phản hồi  
<Hoặc>
- Đặt một nút phản hồi từ bảng **Functions»Structures**



# Khởi chạy Shift Registers & nút phản hồi



# Không chạy Shift Registers & Nút phản hồi



# Tóm tắt

- Có 2 cấu trúc vòng lặp chính: Vòng lặp While và vòng lặp For.
- Tạo trễ hay thời gian cho vòng lặp sử dụng hàm **Wait Until Next ms Multiple**, hàm **Wait (ms)**, hoặc **Time Delay Express VI**.
- Đầu vào vòng lặp là một dạng số học nào thì đầu ra sẽ là một dạng số học đó.
- Feedback nodes và shift registers sẽ truyền giá trị dữ liệu từ một vòng lặp trước tới vòng lặp kế tiếp
- Chỉ sử dụng shift registers khi mà có nhiều phép lặp cần được sử dụng.



# Bài học 4

## Mảng - Arrays

### CÁC CHỦ ĐỀ

Giới thiệu về mảng - Arrays

Tự động chỉ thị mảng

Array Functions

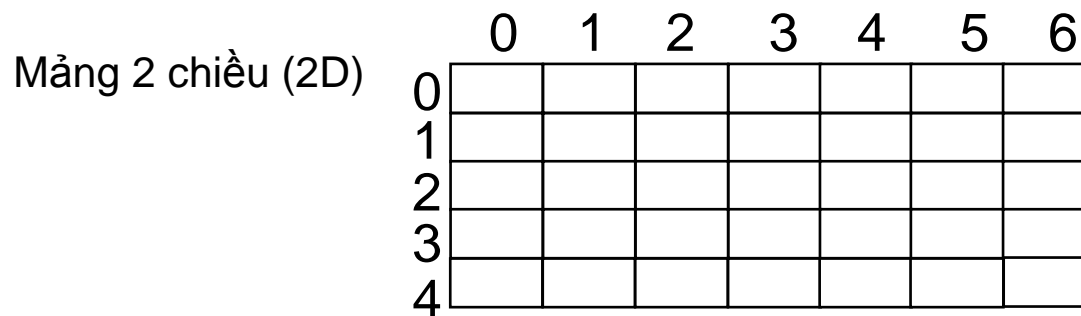
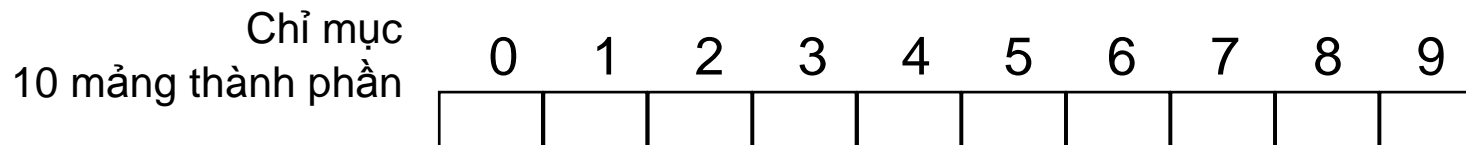
Nhiều mảng kết hợp





# Mảng - Arrays

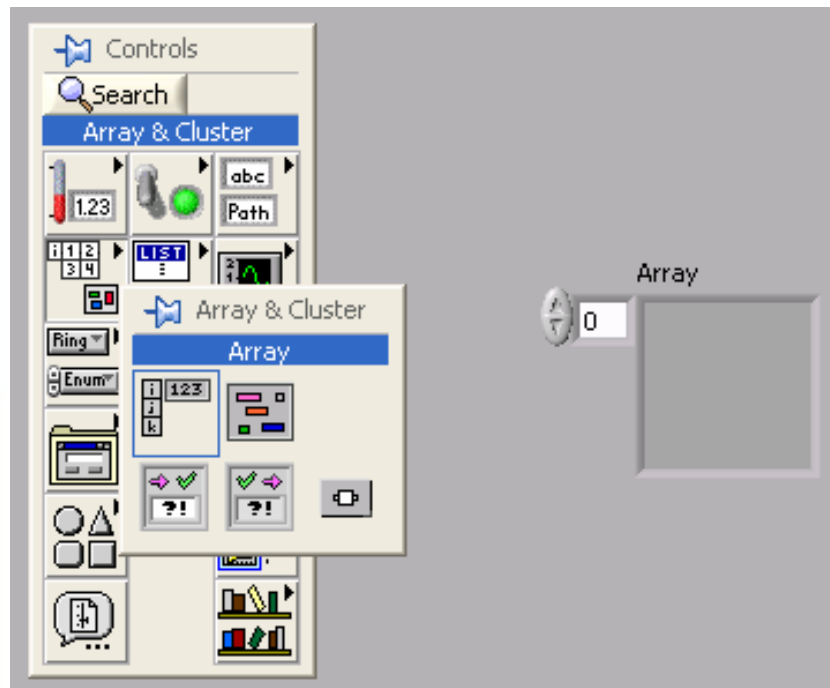
- Tập hợp các thành phần dữ liệu là các dạng tương đương
- Một hoặc nhiều chiều, nhiều hơn 2 thành phần cho mỗi chiều<sup>31</sup>
- Truy xuất các thành phần bởi chỉ mục của chúng; thành phần ban đầu có giá trị chỉ mục là 0



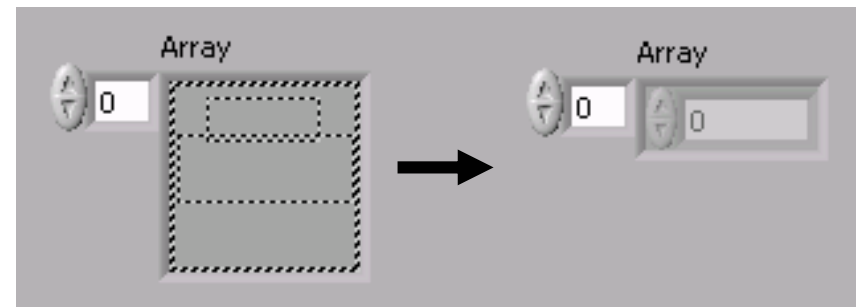
Mảng có 5 dòng, 7 cột có 35 thành phần

# Mảng Điều khiển và chỉ thị

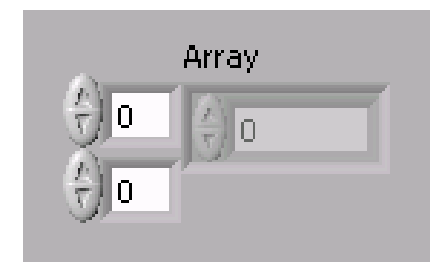
1. Chọn **Array** từ bảng Controls



2. Đặt đối tượng dữ liệu bên trong shell

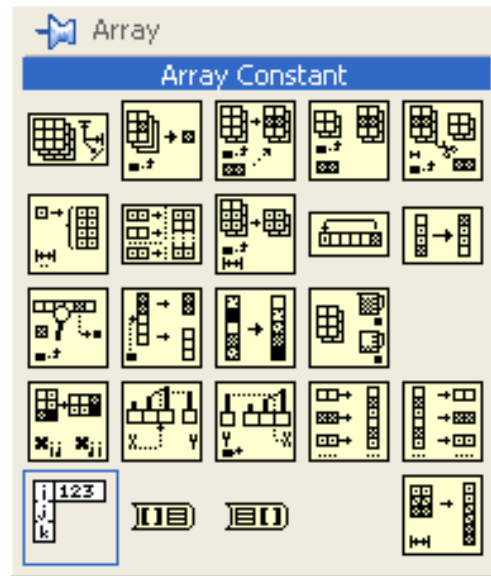


**Add Dimension**  
cho mảng 2D

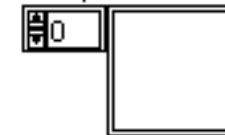


# Tạo mảng hằng số

1. Chọn **Array Constant** từ bảng con của mảng **Array**

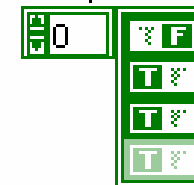


Array Constant



2. Đặt đối tượng dữ liệu vào trong cột mảng đó

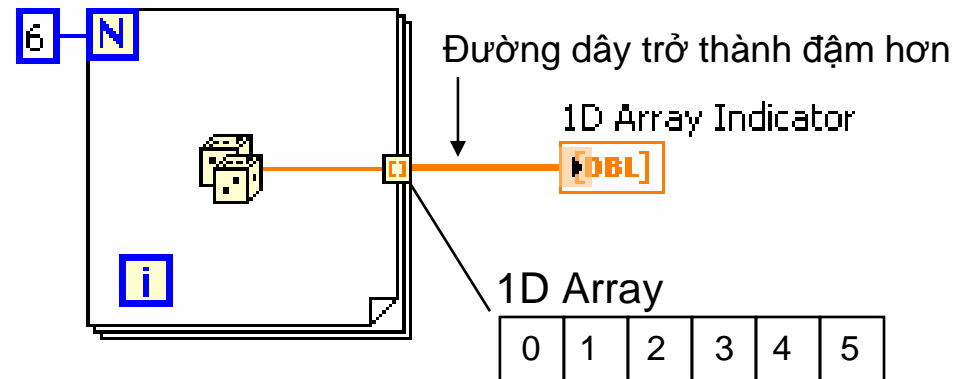
Array Constant



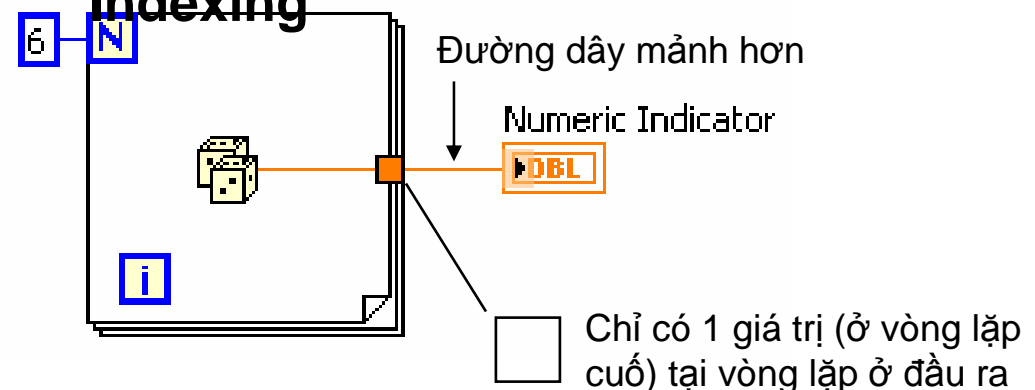
# Tự động chỉ thị - Auto-Index

- Vòng lặp có thể gồm nhiều mảng tự động chỉ thị với đường biên của chúng
- Vòng lặp For tự động chỉ thị mặc định
- Vòng lặp While đưa ra có giá trị kết thúc mặc định.
- Click chuột phải lên tunnel (ô màu vàng nhỏ trên khung) và enable/disable auto-indexing

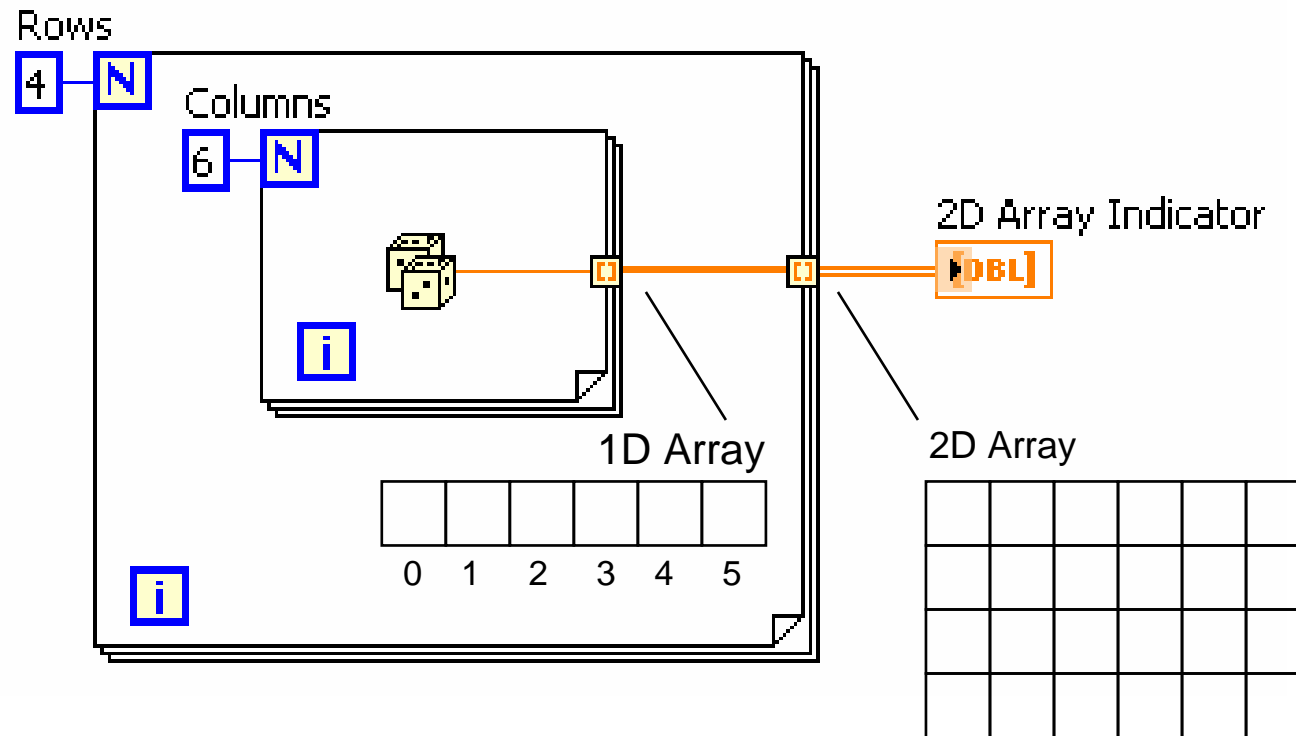
## Cho phép Auto-Indexing



## Không cho phép Auto-Indexing



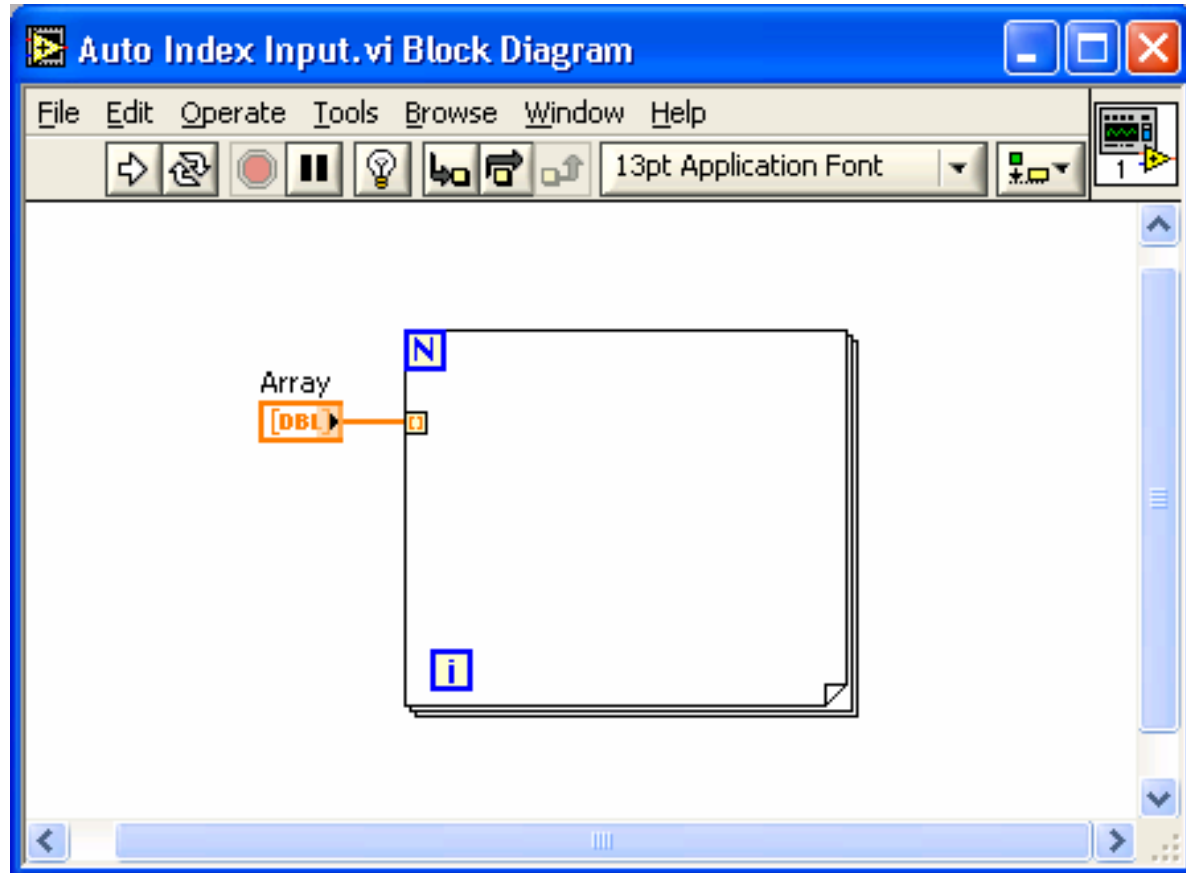
# Tạo mảng 2 chiều - 2D Arrays



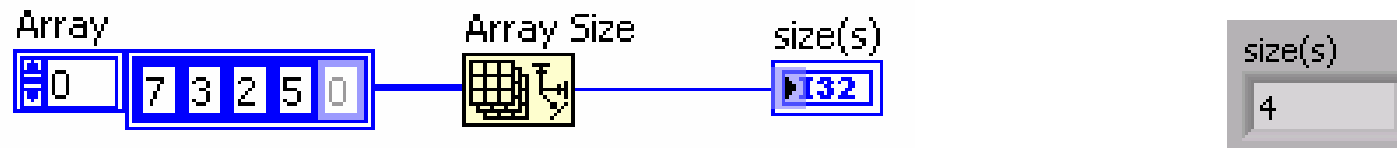
- Vòng lặp bên trong được tạo ra là các cột
- Các ngăn xếp của vòng lặp ngoài sẽ được dây vào dòng

# Đầu vào Auto-Index

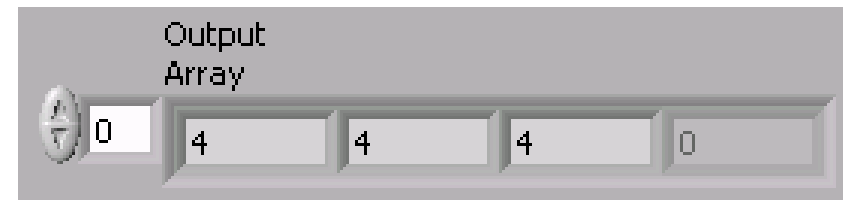
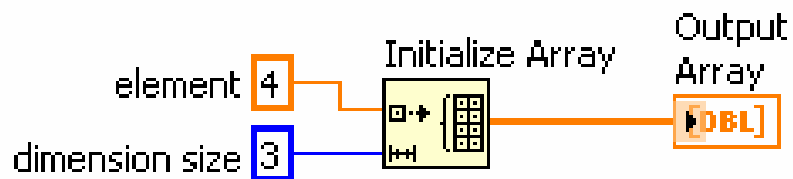
- Một mảng đầu vào có thể được sử dụng để thiết lập vòng lặp For
- Số thành phần trong mảng tương đương với bộ đếm đầu vào
- Mỗi tên nút Run không bị gãy.



# Các hàm mảng phổ biến

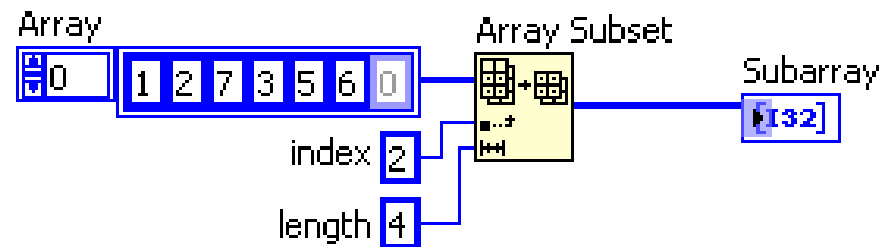


## Cỡ của mảng



## Mảng khởi chạy

# Các hàm mảng phổ biến

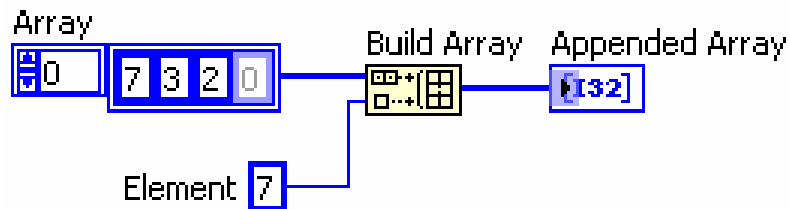


## Tập hợp mảng

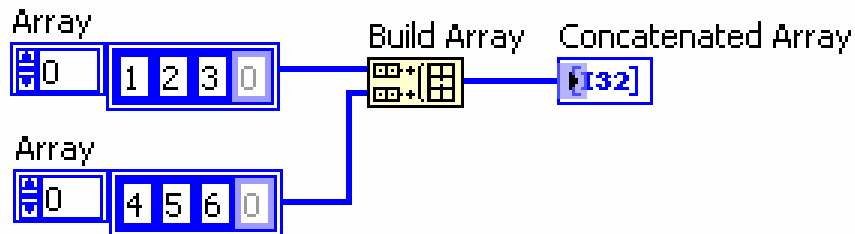




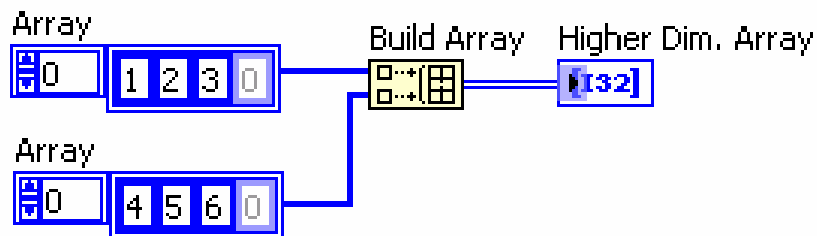
# Xây dựng hàm mảng



## Đưa thêm thành phần dữ liệu



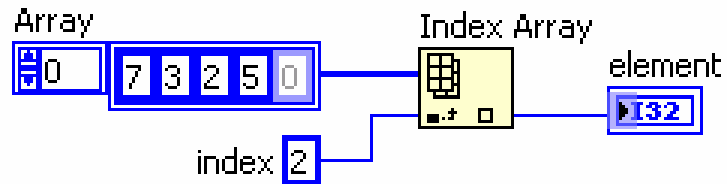
## Đầu vào nối tiếp nhau



## Xây dựng một mảng nhiều chiều hơn

*Mặc định*

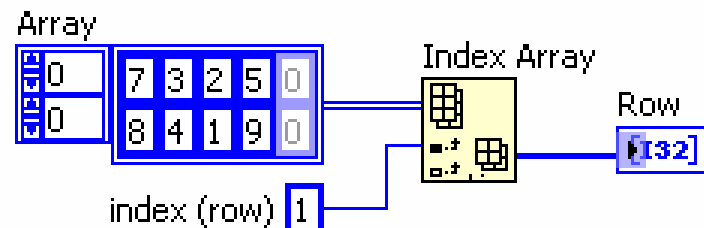
# Hàm Index Array



element

2

## Trích 1 thành phần

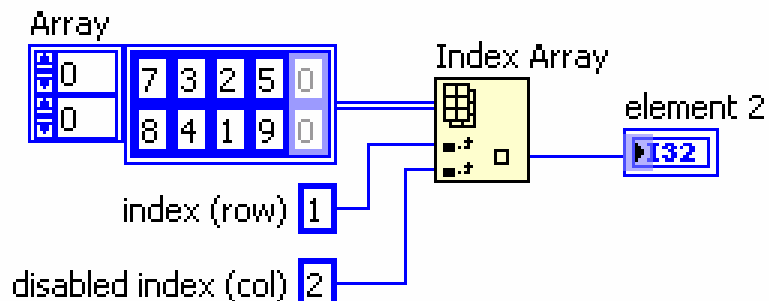


Row

0

8 4 1 9 0

## Trích 1 dòng



element

1

## Trích thành phần của dòng

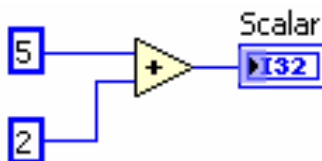
# Các dạng khác

Những hàm đầu vào có thể có kiểu khác nhau

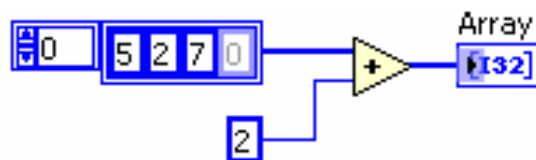
Các hàm số học của tất cả phiên bản LabVIEW đều có nhiều dạng khác

## Kết hợp

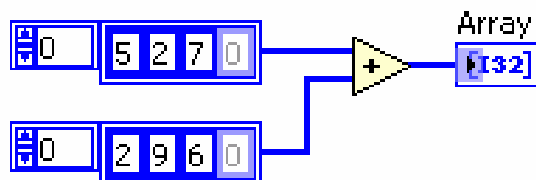
Vô hướng + Vô hướng



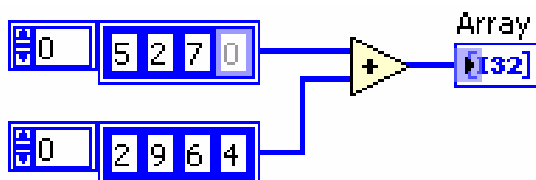
Mảng + vô hướng



Mảng + mảng

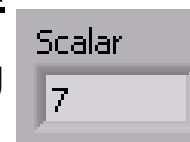


Mảng + mảng



## Kết quả

Vô hướng



Mảng



Mảng



Mảng



# Tóm tắt

- Nhưng thành phần dữ liệu trong nhóm mảng đầu có kiểu tương đương. Bạn có thể xây dựng nhiều mảng: numeric, Boolean, path, string, waveform, và kiểu dữ liệu cluster.
- Mảng chỉ thị cơ sở là 0, có nghĩa là nó có dải từ 0 tới  $n - 1$ , trong đó  $n$  là một số trong mảng.
- Để tạo 1 mảng điều khiển hoặc chỉ thị, ta lựa chọn một mảng ở bảng sau: **Controls»Array & Cluster**, thay thế nó trong Front panel, và kéo thả đối tượng điều khiển hoặc chỉ thị vào cấu trúc mảng.
- Nếu bạn đi dây một mảng đầu vào cho vòng lặp For hoặc While, bạn có thể đọc và quy trình cho các thành phần trong mảng bằng cách enable auto-indexing.
- Mặc định ban đầu, LabVIEW là enable auto-indexing trong vòng lặp For và disable auto-indexing trong vòng lặp While.
- Các dạng khác tùy thuộc vào dữ liệu đầu vào của cấu trúc dữ liệu.

