LabVIEW base course

Dott. Mirco Andreotti – INFN Ferrara
May 2018

Time Table

- Mer 09/05 14:30 - 16:30 AULA 017
- Gio 10/05 9:30 - 12:30 AULA 017

- Mar 15/05 9:30 - 12:30 AULA 100
- Mer 16/05 14:30 - 16:30 AULA 017

- Mar 22/05 9:30 - 12:30 AULA 100

Outline

- Lessons 1-2

1. (Labview installation)

2. Labview environment

3. Variables, functions, base features, write/read files

4. Dataflow, parallelism and time syncronization

- Lessons 3-4

5. Data Acquisition (DAQ) structure

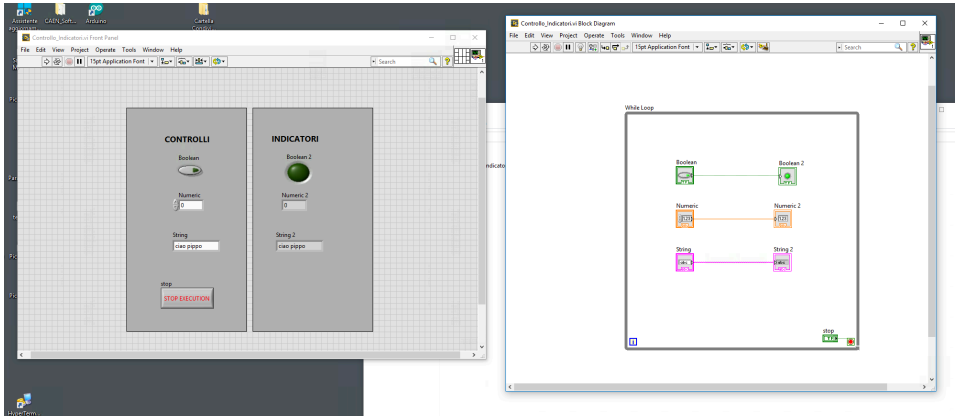
6. Labview interface with external hardware
-- communication type and protocol
-- examples with Arduino microcontroller
-- examples in our laboratories

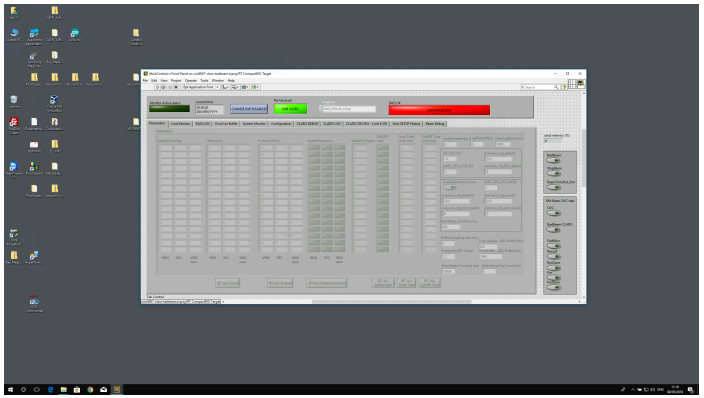
- Lessons 4-5

7. Case study memory-consuming
8. Brief intro to DAQ with National Instruments hardware
9. Brief intro to Labview - FPGA
10. Extra topics

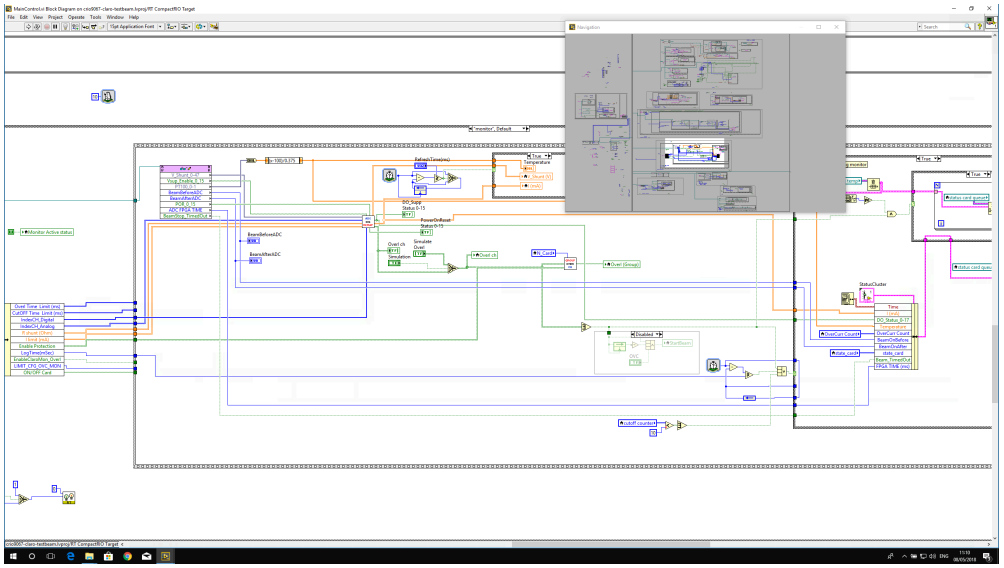
WHY LABVIEW?

- License costs some kEuros
- Fast implementation of user interface/code
- “Easy” interface with “all” laboratory hardware
- Intuitive graphical programming
- A lot of ready tools (remote control, user interaction, protocols …)





- ➔ Ideal for laboratory setup
- ➔ Ideal for standalone applications
- ➔ Ideal for large project

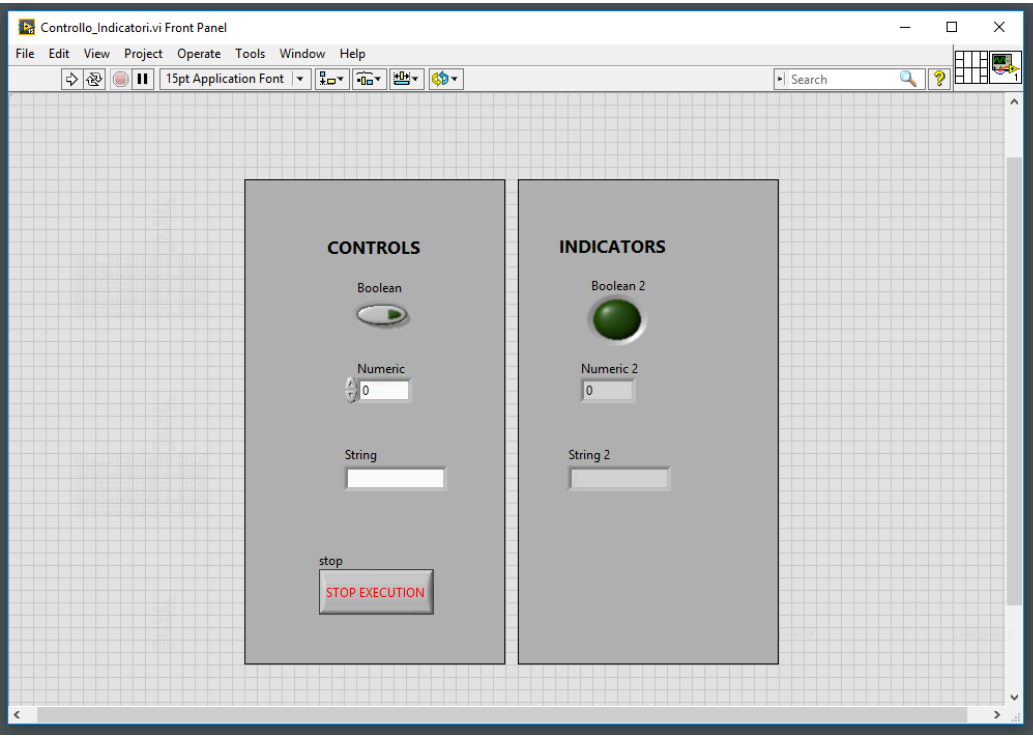


- ✓ Labview applications need detailed design
- ✓ Take care of memory consuming
- ✓ Take care of time alignment
- ✓ Risk of a “divergent” code

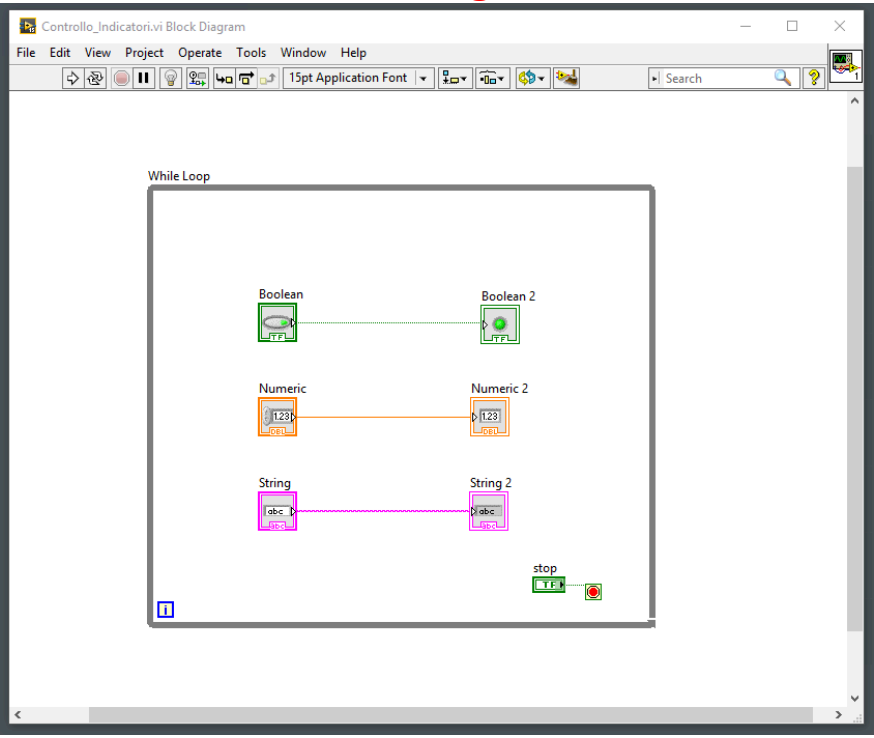
LabVIEW environment

- ✓ Labview files extention: .vi (Virtual Instruments)
- ➢ Labview VI consists in two parts:
  - ❑ Front Panel → GUI (Graphical User Interface)
  - ❑ Block Diagram → Code

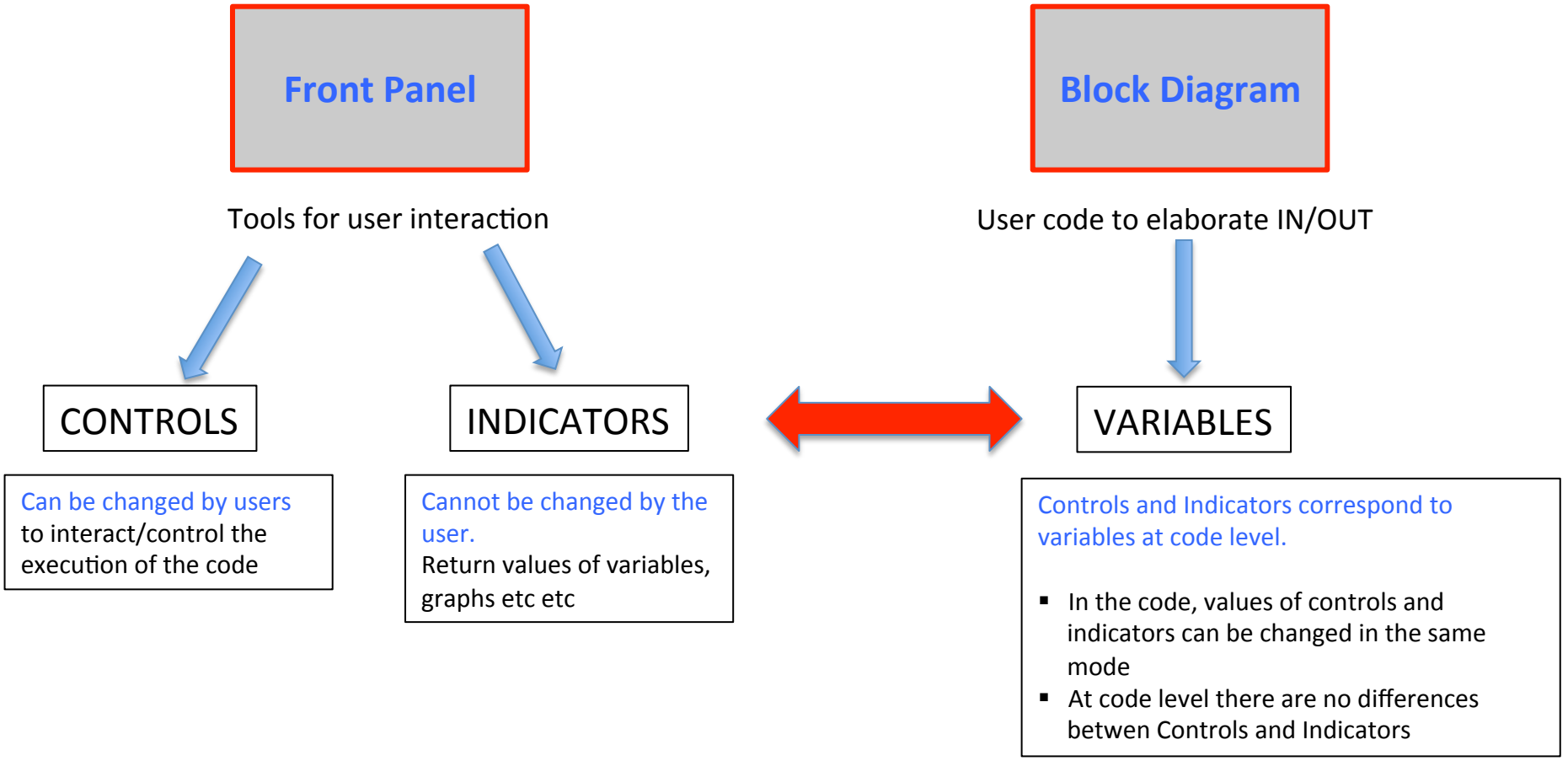
Front Panel



Block Diagram



LabVIEW environment



LabVIEW environment

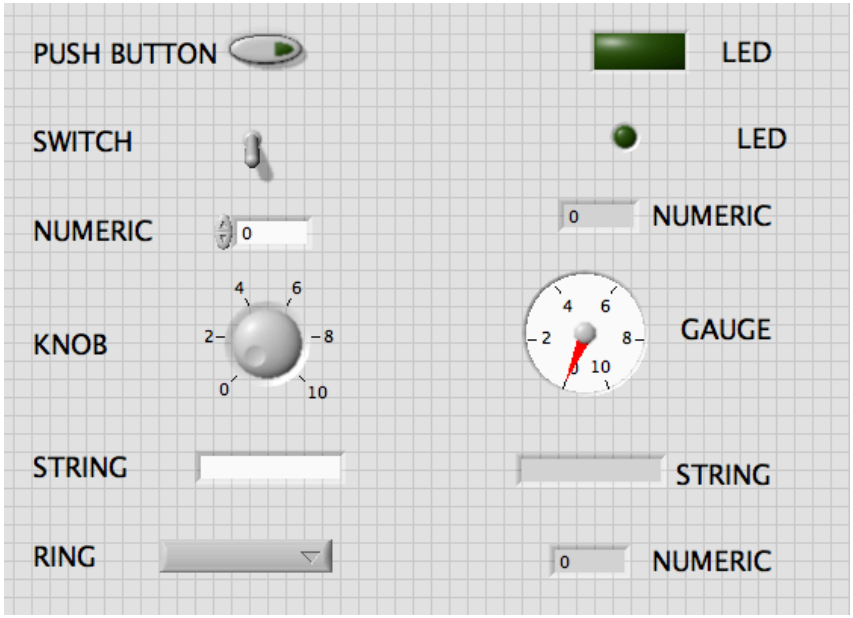
Front Panel

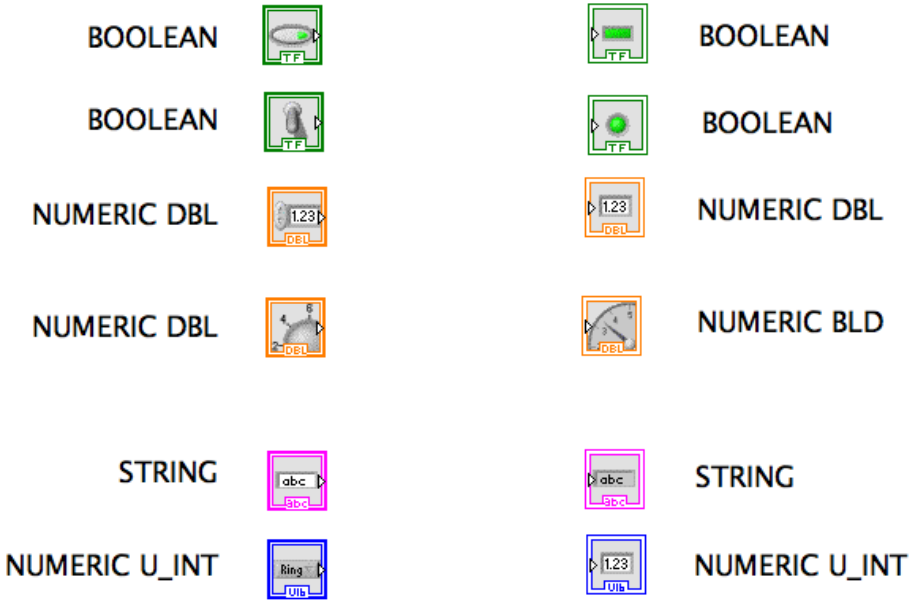
Block Diagram

CONTROLS

INDICATORS

VARIABLES

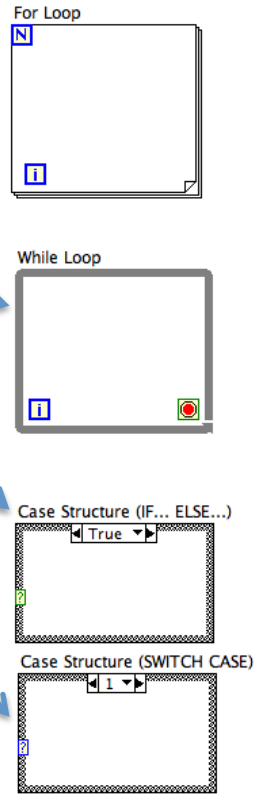




Block Diagram programming tools

- **VARIABLES**
  - Boolean
  - Numeric
  - String
  - Timestamp (cluster)
  - Cluster (like C structure)
  - Array of (boolean, numeric, string, cluster…)
  - …

- **CONTROL FLOW ELEMENTS**
  - FOR LOOP
  - WHILE LOOP
  - CASE STRUCTURE
    - IF … ELSE …
    - SWITCH CASE
  - …
  - EVENT HANDLER
  - …

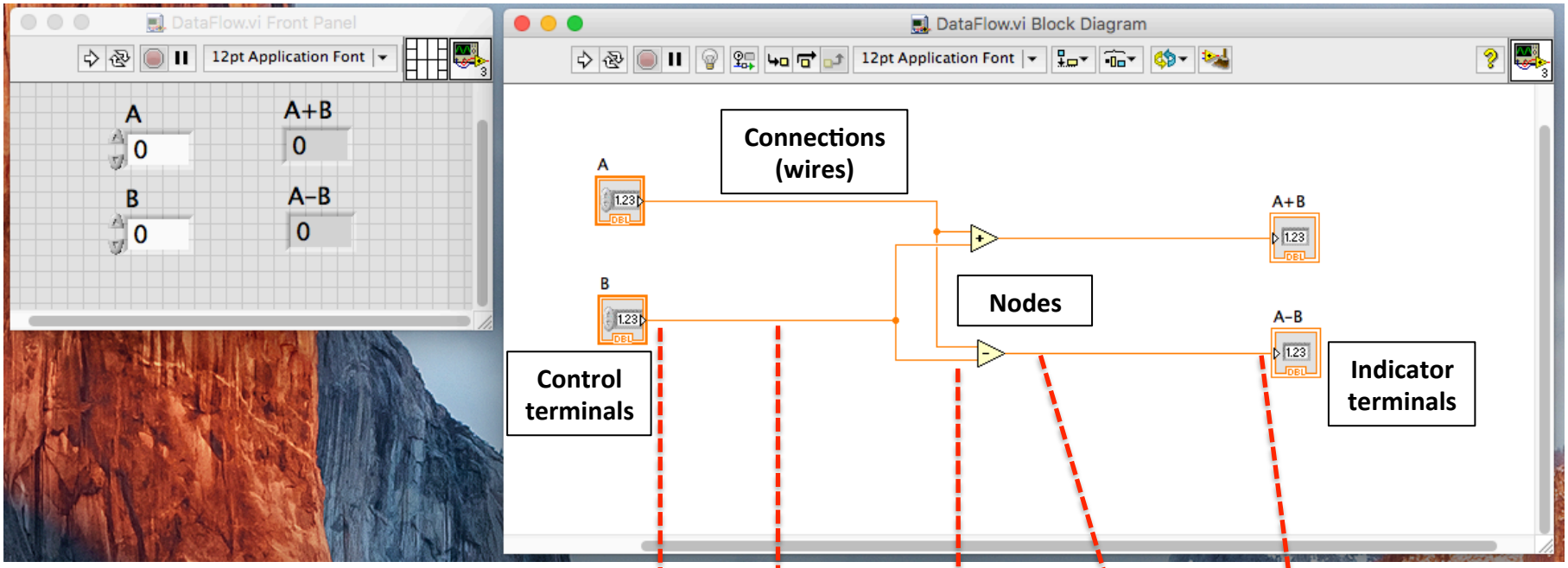


- **MATHEMATICAL TOOLS**
  - Base operations
  - Formula nodes
  - …
- **STRING MANIPULATION TOOLS**
- **DATA ANALYSIS TOOLS**
  - Statistics analysis
  - Fitting
  - …
- **SIGNAL PROCESSIONG**
  - FFT
  - …
- **HARDWARE INTERFACE TOOLS**
  - Serial comunication
  - Ethernet communication
  - Hardware specific interface
  - Industrial protocol
  - …
- **… ALL A PROGRAMMER NEEDS ….**

Block Diagram code execution: DATA FLOW

- Base elements in a block diagram

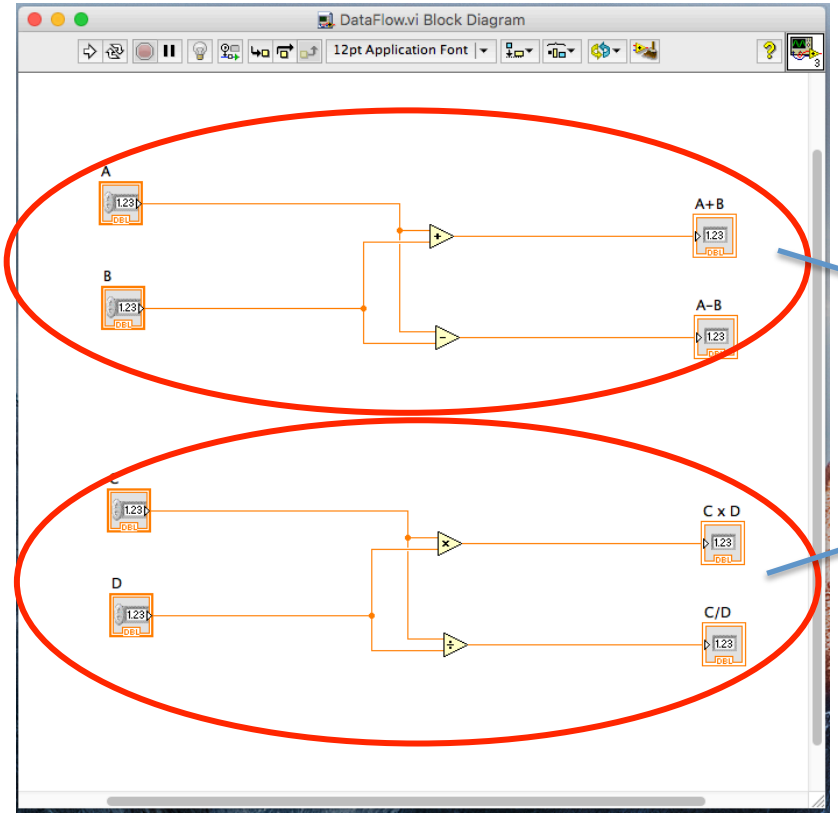
- Control/Indicator terminals (something like a declaration of variables)
  - Control terminal emits data-value (output)
  - Indicator terminal receives data-value (input)
- Nodes : elaborate the INPUTS to produce OUTPUTS
- Connections : tells data path to follow from OUTPUTS to INPUTS



Code execution follows DATAFLOW model

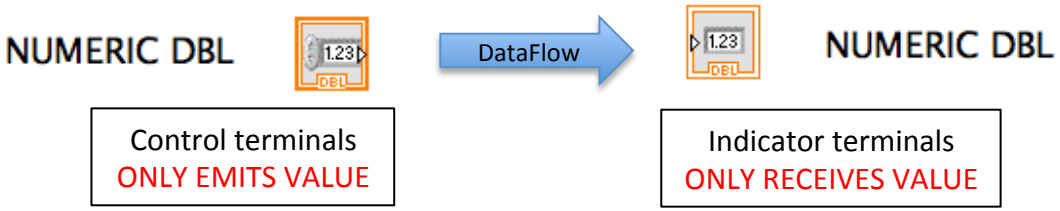
Block Diagram code execution: DATA FLOW

- DATAFLOW model (some) features
  - ✓ A node is executed only when all data reach it
  - ✓ Allow to distribute data in parallel
  - ✓ Allow parallel execution of different pieces of code



Parallel execution

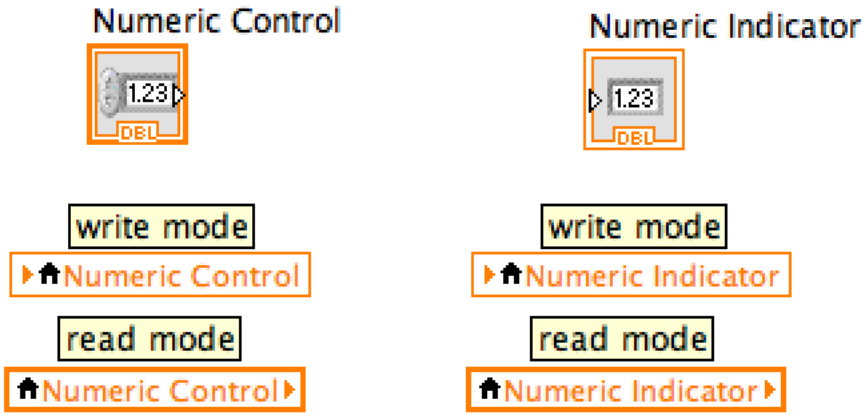
Block Diagram variables



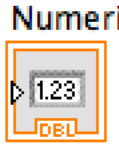
✓ VERY LIMITING the use of only terminals and connections

➔ LOCAL VARIABLES

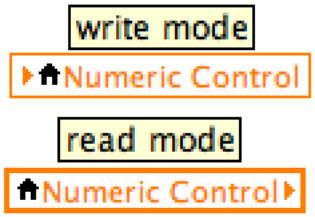
- Allow to call variables inside code (like in C you call variable by name)
- Local variable can be in READ and/or WRITE mode
  - → Local variable of a Control Terminal allow to change control value inside code
  - → Local variable of a Indicator Terminal allow to use the value inside code
- Allow to create clean block diagram of the code

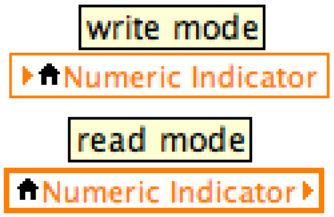


Block Diagram variables



➔ “Declaration” of the variables



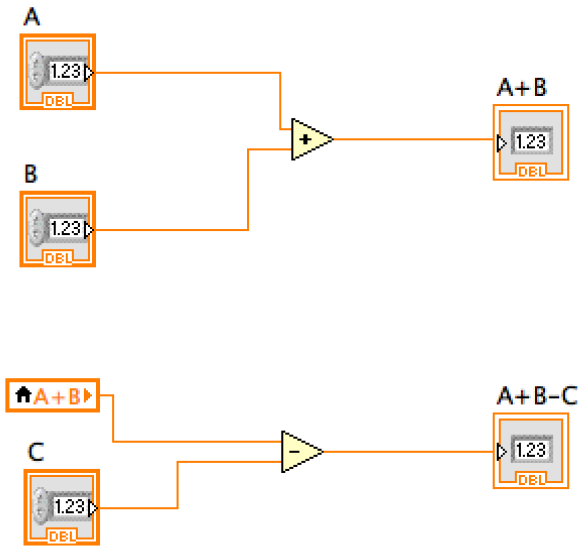


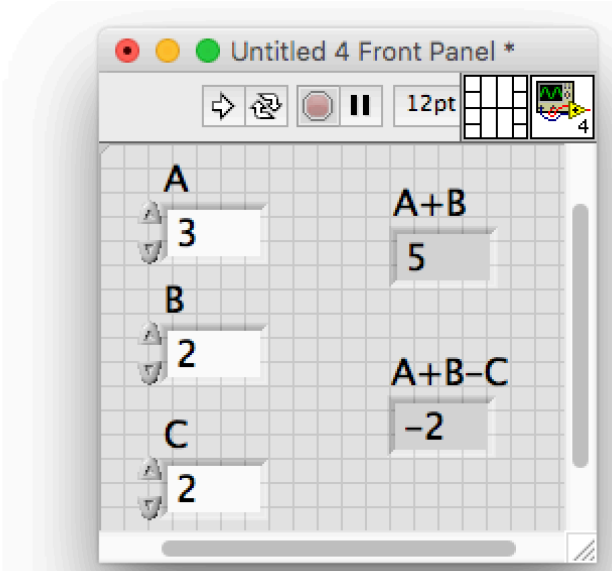
➔ Call of variable to write on it

➔ Call of variable to read it

!!! Pay attention at parallel read/write of local variables

➔ Users need to handle time alignment





???

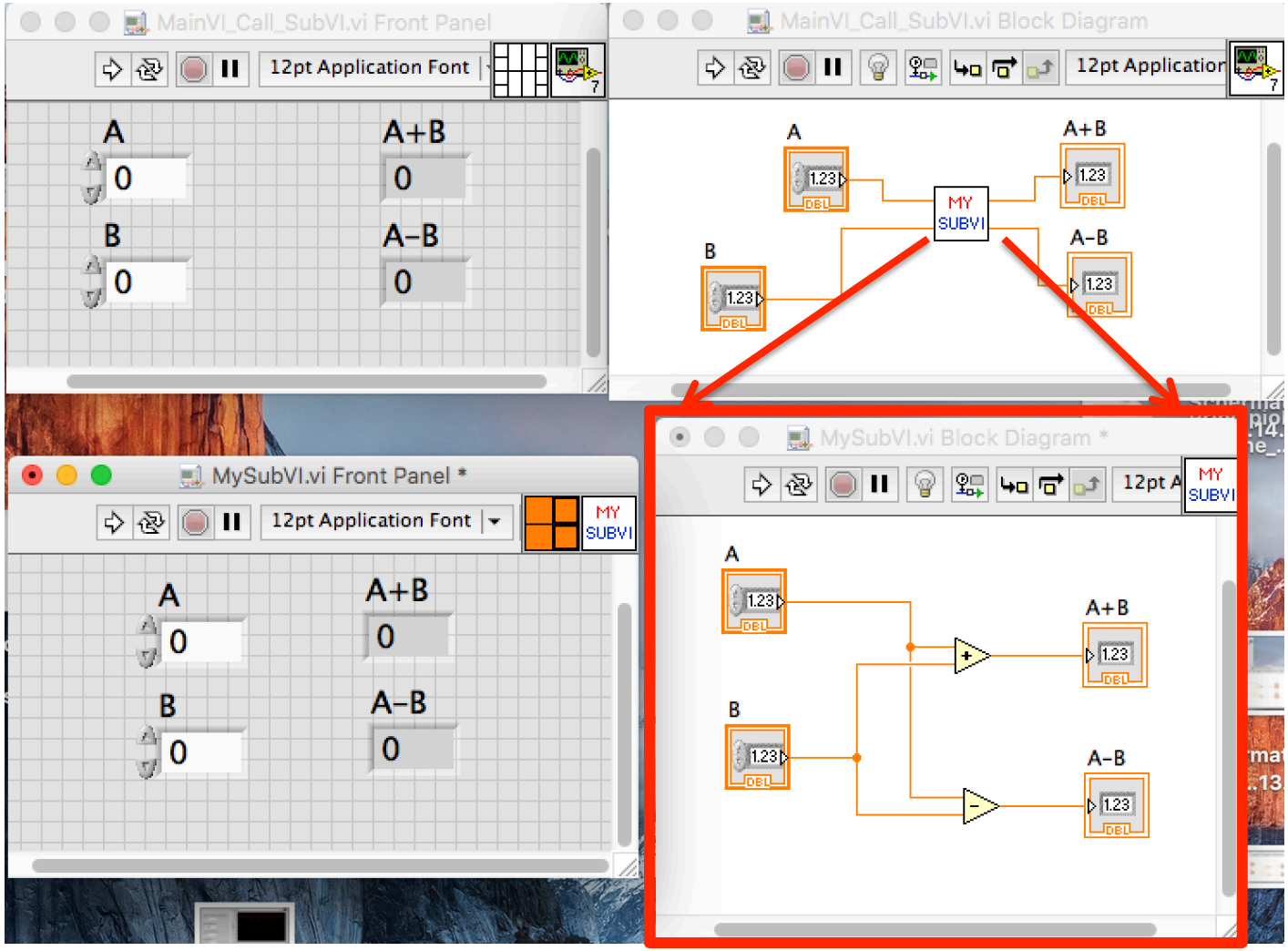
Other variables

➔ GLOBAL VARIABLES
- To share values between different VIs running on same PC

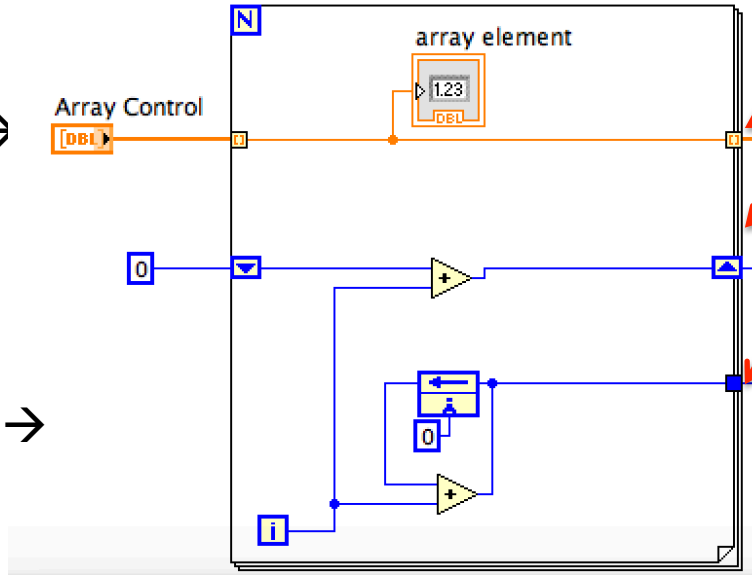
➔ NETWORK SHARED VARIABLES
- To share values between different Vis running on different PCs connected on network
- Some network protocol is implemented and handled by labview

SubVI

VI configured with INPUTS and OUTPUTS and runs inside other main VI



FOR/WHILE LOOP useful features



Tunnel mode

Indexing of array →

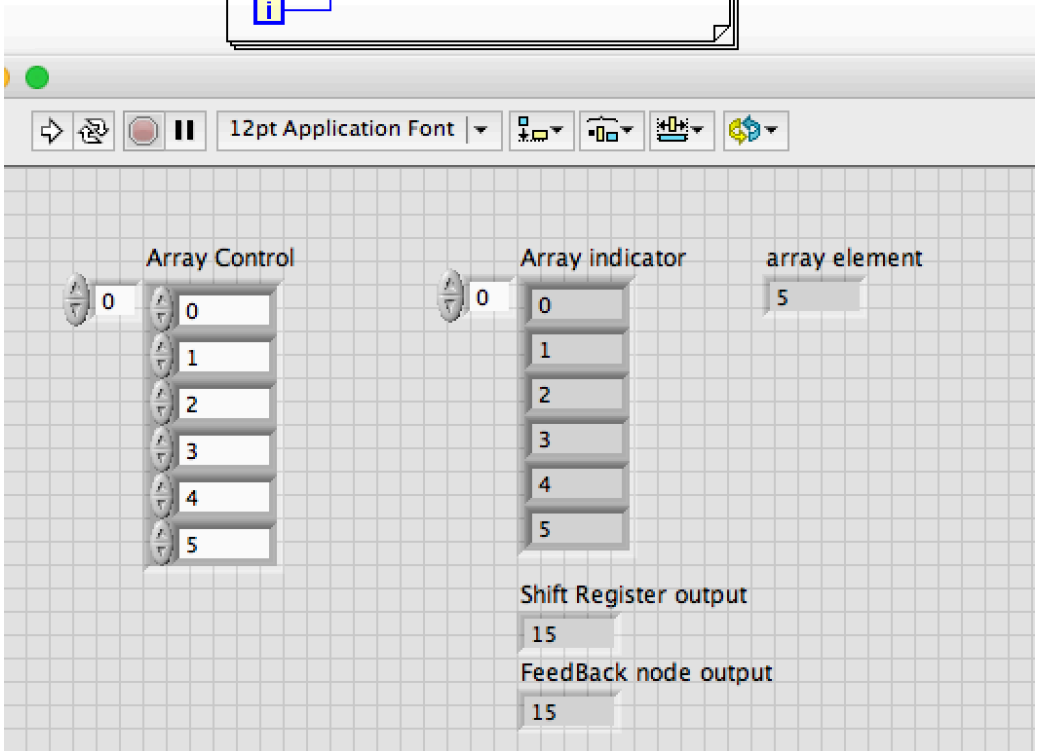
→ Indexing of array

Shift register →

→ Shift register out

Feedback node →

→ Last value



Other tunnel modes:

- Concatenating
- Conditional

Read/Write Files

- Binary file
  - Need to know bit-format of data
  - Useful for high speed write
  - Need post-processing
- Text file
  - Define a format of data
  - Write: convert numeric into string e compose the defined format
  - Read: from defined format read string, parse the string and convert single piece into number
  - Useful for human-readable info
  - Usefule for analize data with external software
  - Usually used for no post-processing
- Datalog file
  - Datalog is a labview- formatted file type
  - Read/Write data Cluster
  - Need to know tha Cluster composition
  - Need post-processing for use with other external software
  - Ideal if files are used only inside labview
- Spreadsheet file
  - Useful for array and matrix
- TDMs file
  - Specific for waveforms