Essentials of Crestron Programming

Training manual - 2005

Instructors: Steven Schoeters
Jan Ooms
Charles Hazelhoff

Program Overview

1. Installation of all necessary software.
2. Establishing communication with the control system.
    A. Viewport
    B. Toolbox
3. What is SIMPL windows?
4. Signals and symbols
5. The Configuration and Programming screen – where to find everything
6. 5 exercises which will guide you into SIMPL windows programming
7. Touchpanel design with VTPRO-e and layout activation with SIMPL Windows
    A. Clock/date activation
    B. Subpages
    C. Serial indirect text
    D. Analog control
    E. Animation
    F. Modules
    G. Serial
    H. Testmanager
    I. Signal routing
8. Assignment

1. Installing necessary software:

Training CD contains: SIMPL Windows, VTPRO-e, database, manuals, datasheets, examples,…

- Crestron Software is officially compatible with Windows 98SE/2000/NT and XP

- What to install:

  SIMPL windows

  VTPRO-e

  SIMPL windows library (containing all crestron products)

  Crestron database

  The Crestron database is a collection of information that is accessed by various Crestron software packages, including SIMPL Windows, VisionTools Pro, VisionTools for Windows, and DEAL for Windows.

  SIMPL Windows Cross compiler

  Converts the program into machine code language

- Make sure you keep everything up to date!

  Check the following link for updates:
  http://www.crestron.com/downloads/software_updates2.asp

2. Establishing communication

Requirements

- Viewport software (included in SIMPL windows and VTPRO-e) or Toolbox (separate installation)

- Straight or crossed cable (depending on the control system)

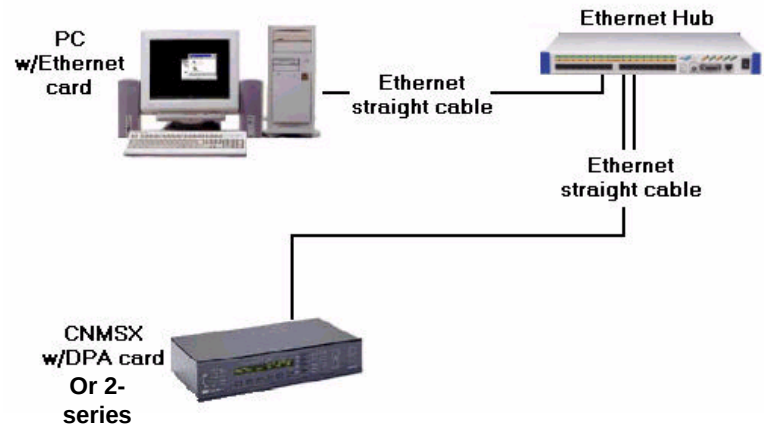
It is possible to connect in 2 different ways:

RS-232 Connection

For RS-232, use a DB9 straight-through or crossed serial cable to connect the COMPUTER port on the control system to one of the COM ports on the PC.

TCP/IP Connection

For a TCP/IP connection, use Ethernet straight cables to connect the PC and control system to the LAN.



A. Viewport

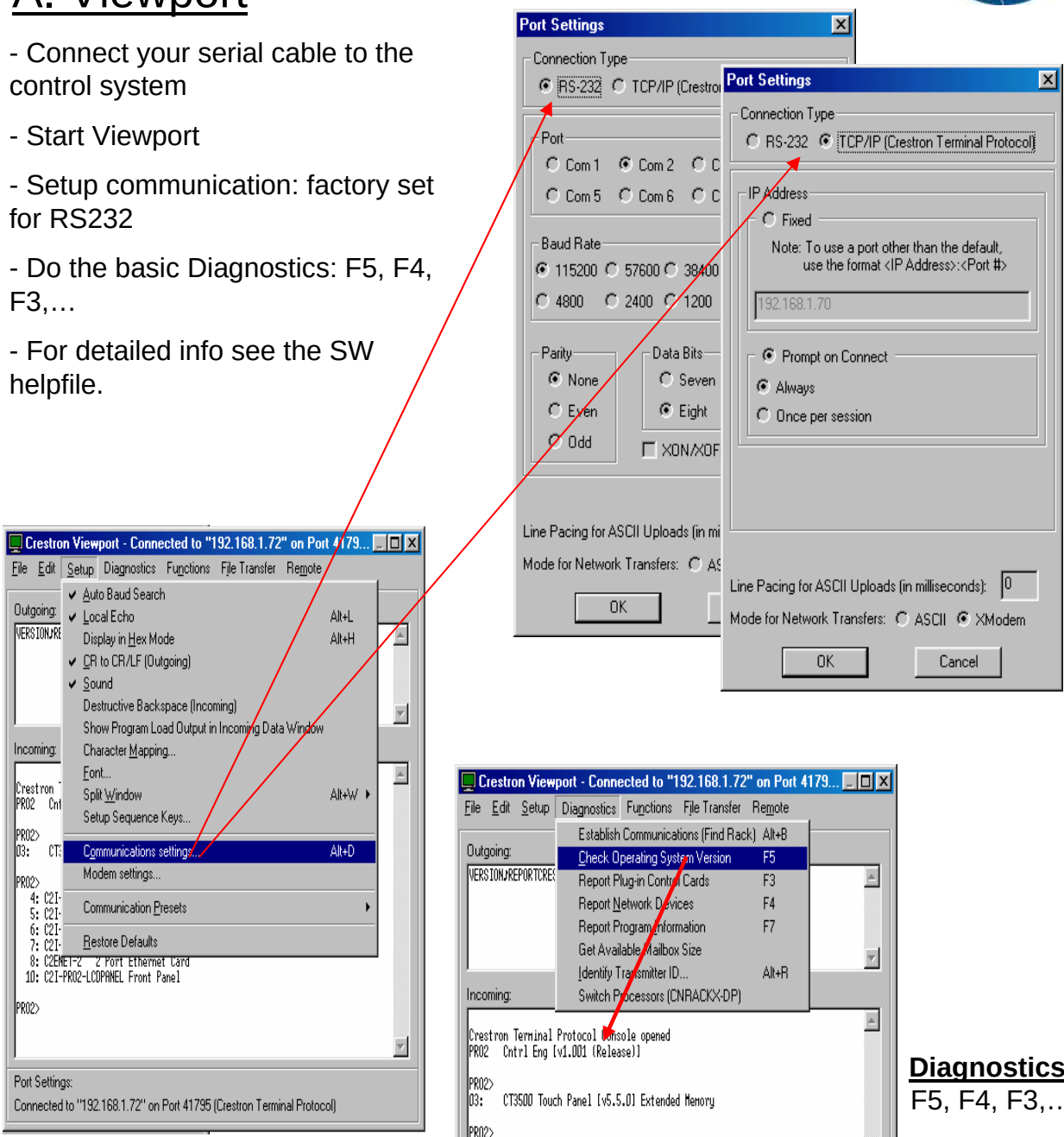
- Connect your serial cable to the control system

- Start Viewport

- Setup communication: factory set for RS232

- Do the basic Diagnostics: F5, F4, F3,…

- For detailed info see the SW helpfile.

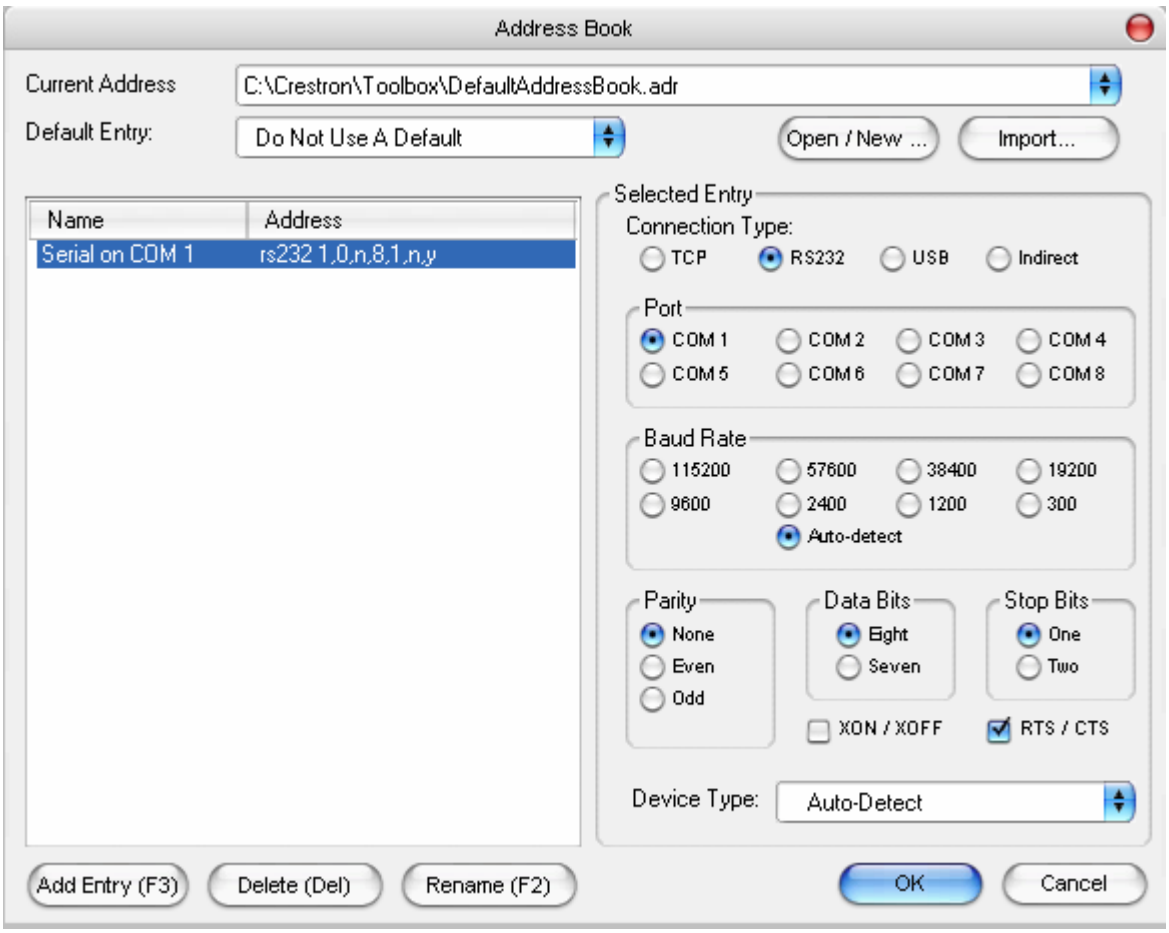


Diagnostics:
F5, F4, F3,…

B. Toolbox

The Crestron Toolbox is intended to fully replace the Crestron Viewport.

Viewport's general architecture is to connect to a control system and then perform functions from the command line. With the Toolbox, this has been replaced by easy-to-read graphical interfaces and the ability to connect discretely to specific devices from each tool.



The Address Book allows you to maintain a list of devices that can communicate with the PC. The addresses are saved in an .adr file, allowing you to easily share address books with other programmers, edit or remove addresses, and import addresses.
You can establish a session with any device by selecting the address from the MRU (most recently used) address list in the status bar at the bottom of each system tool.

The Crestron Toolbox provides the following tools:

Text Console: Performs text-based (command-line) functions.

SMW Program Tree: Lists devices in a SIMPL Windows program. Allows you to update firmware, verify devices, upload projects and link to the Network Device Tree.

Network Device Tree: Lists devices detected on the network. Allows you identify devices, manage device Network IDs and link to the SMW Program Tree.

Script Manager: Runs scripts for automating system tasks.

System Info: Displays general device information. Allows you to manage device functions and capture debugging information.

File Manager: Displays the file system. Allows you to manage control system files and directories.

Network Analyzer: Samples voltages on the Cresnet Y and Z wires. Allows you to troubleshoot Cresnet network problems.

Video Test Patterns: Generates test patterns for calibrating video.
The Crestron Toolbox allows you to perform these functions using simple graphical views and click and drag methods.

3. What is SIMPL windows?

SIMPL Windows is an abbrevation of:

Symbol

Intensive

Master

Programming

Language

SIMPL Windows combines the familiar drag-and-drop functionality of Microsoft Windows with programming power. It provides the link between Crestron systems hardware, users interfaces, and the world of equipment to be controlled.

It is a tool that lets you configure, program, test and debug an integrated control system application.

4. Signals and symbols

A **symbol** is an item with a certain **logical function**, it can manipulate a “signal” in many different ways: pulse, stretch, delay, keep high, keep low, set value, change value, …

Symbols (hardware and software) are interconnected with “**signals**”:

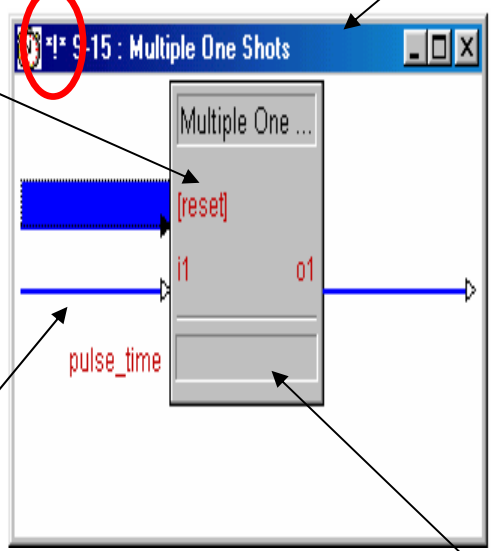
- Digital: High/Low – 1 or 0 = **Blue**

- Analog: Value between 0 and 100% (corresponds with analog channel) = **Red**

- Serial: Data (text,…) **Black**

“**F1**”

Functions “with” [brackets] do not have to be used, they are optional and can be used when required



Pressing “**F1**” on a selected symbol will give you a complete detailed description. (see also SW manual on the Control CD)

Functions “without” brackets have to be used, otherwise you will get an error message ***!* Incomplete Symbol** (required in or output is missing) and the symbol will malfunction or NOT at all.

“Parameters have to be filled in, otherwise you will get an error message ***!* Incomplete Symbol** (required in or output is missing), and the symbol will NOT function

The first 5 exercises are done on this standard Training screen. (VTPRO-e is only used the 2nd day.)

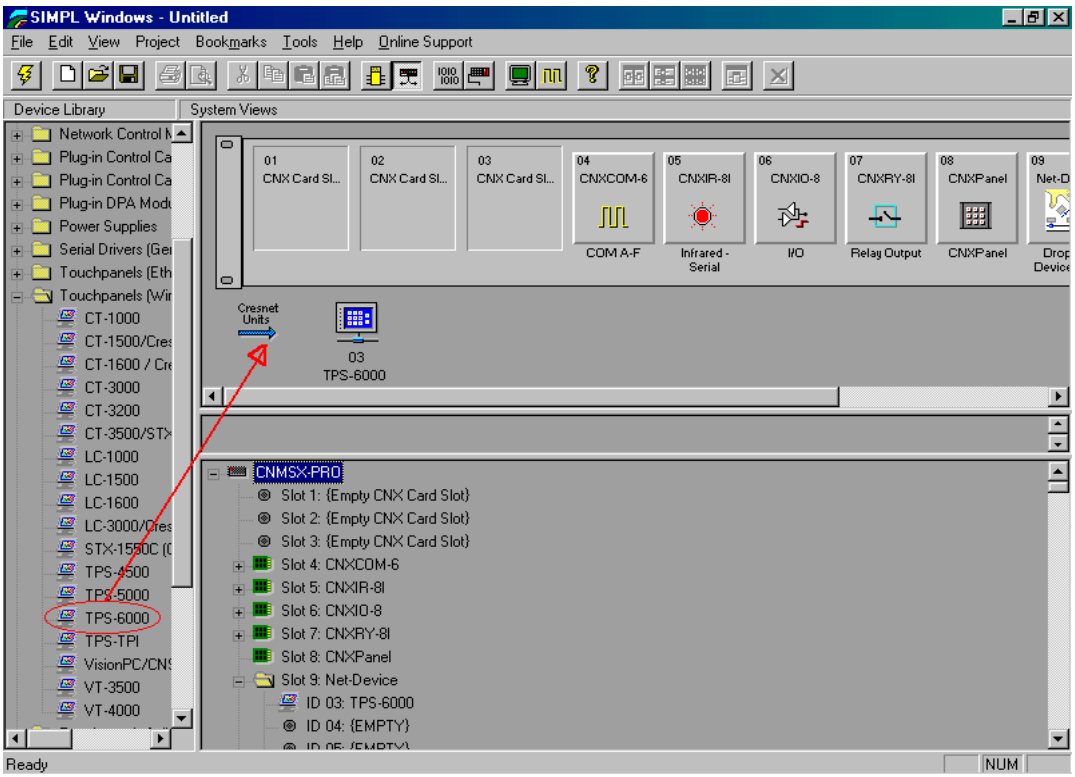
The numbers on the buttons correspond with the "**JOIN**" numbers of the buttons.

The **Join number** is a reference number of these buttons in SW, it "Joins" (connects) the button on the TP layout with the button/TP definition in the SW program.

5. Configuration view

- To configure a system select items from the Device Library and drop them in the card slots/network.

- Difference between the CRESDB (Crestron Database) and the USERDB (User database)



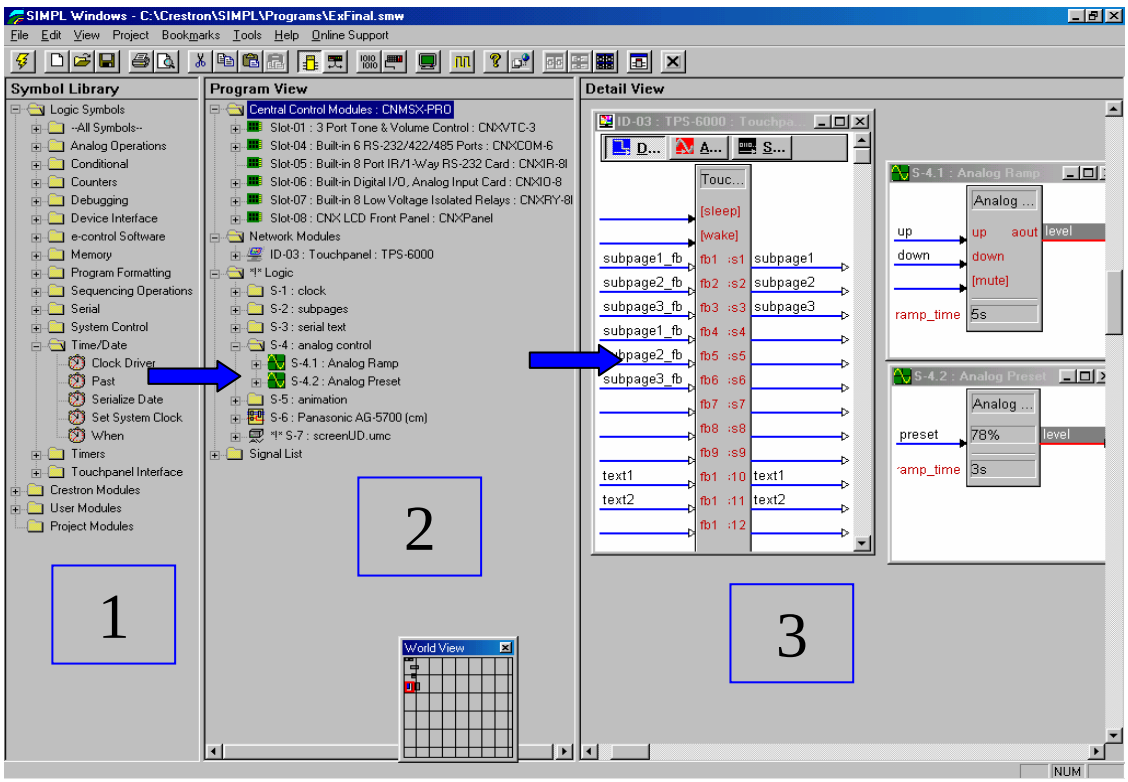
5. Programming view

1- Windows folder structure – under **Program View**

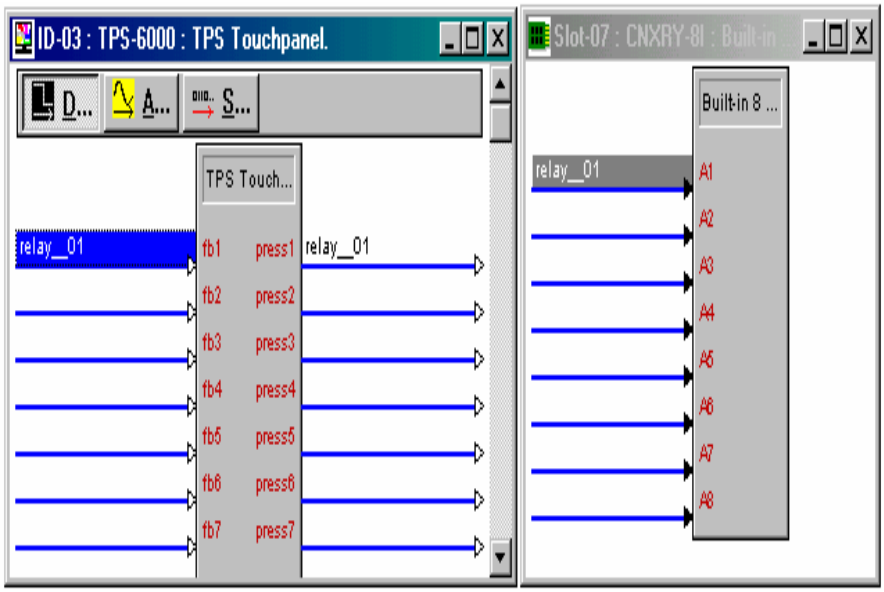
2- **Symbol Library** – where the building blocks (symbols, modules) are located

3**- Detail View:** detail view of the main structure under program view

4- Drag symbols/modules out of the **Symbol Library** and place them in the **Program View**, then take the required symbols from the program view and drag them in the detail view to make the required signal connections



6. Exercises



Direct control:

- Digital signal

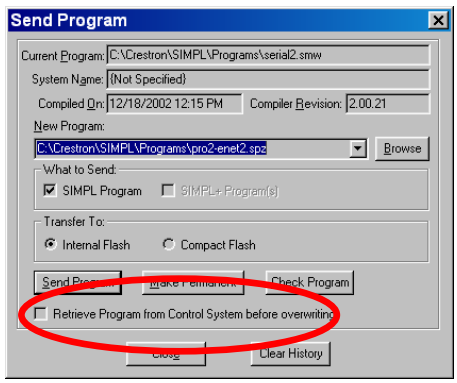
- Signal goes direct from TP to relay module, no logic symbols in between.

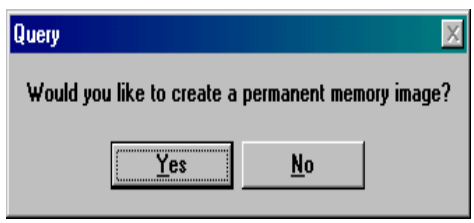
- Connection is made by giving out- and input the same name

- F2, F3 signal routing,…“shortcuts”

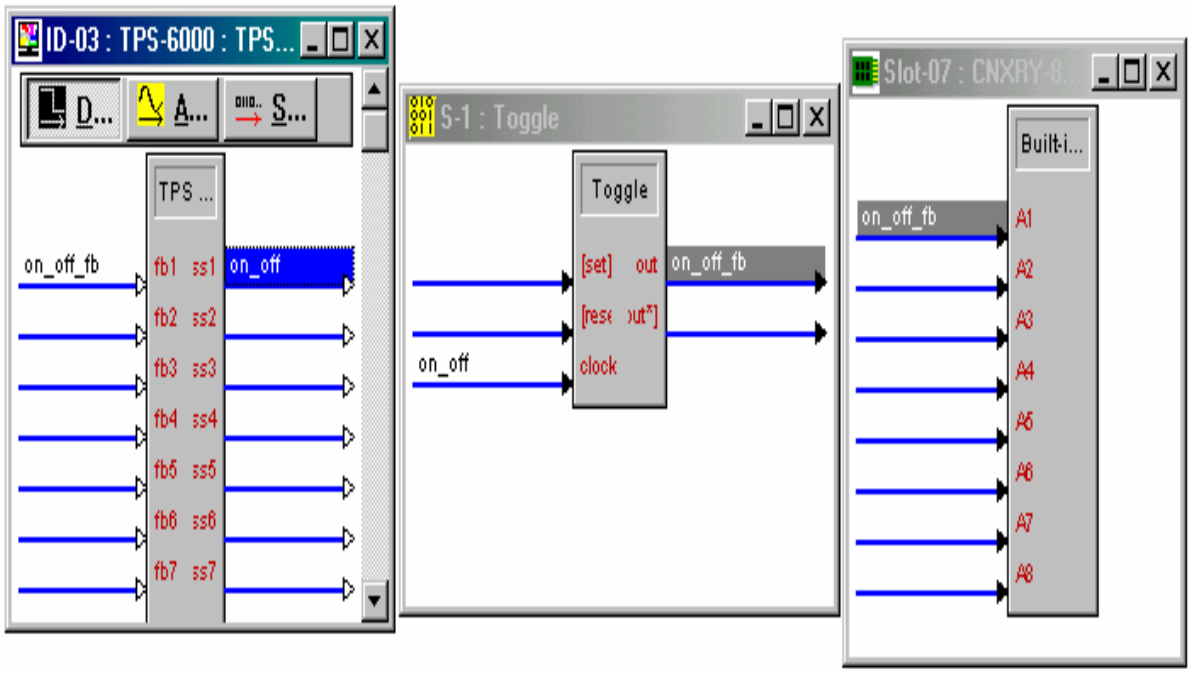
- **CNX series:** “**Permanent Memory Image**”. In case you are not sure of the new program do not do it untill you tested it. In case it is not good, you can get the old program back by re-booting the system.

- **2-Series**: After an upload the “**Permanent Memory Image**” is done automatically, but you have the chance (splash screen) to make a backup of the running program on your PC before you upload. In case the new program is not OK you can than still reload the old program so the user can continue working with the system.





Example 2



Indirect control **(via logic symbols):**

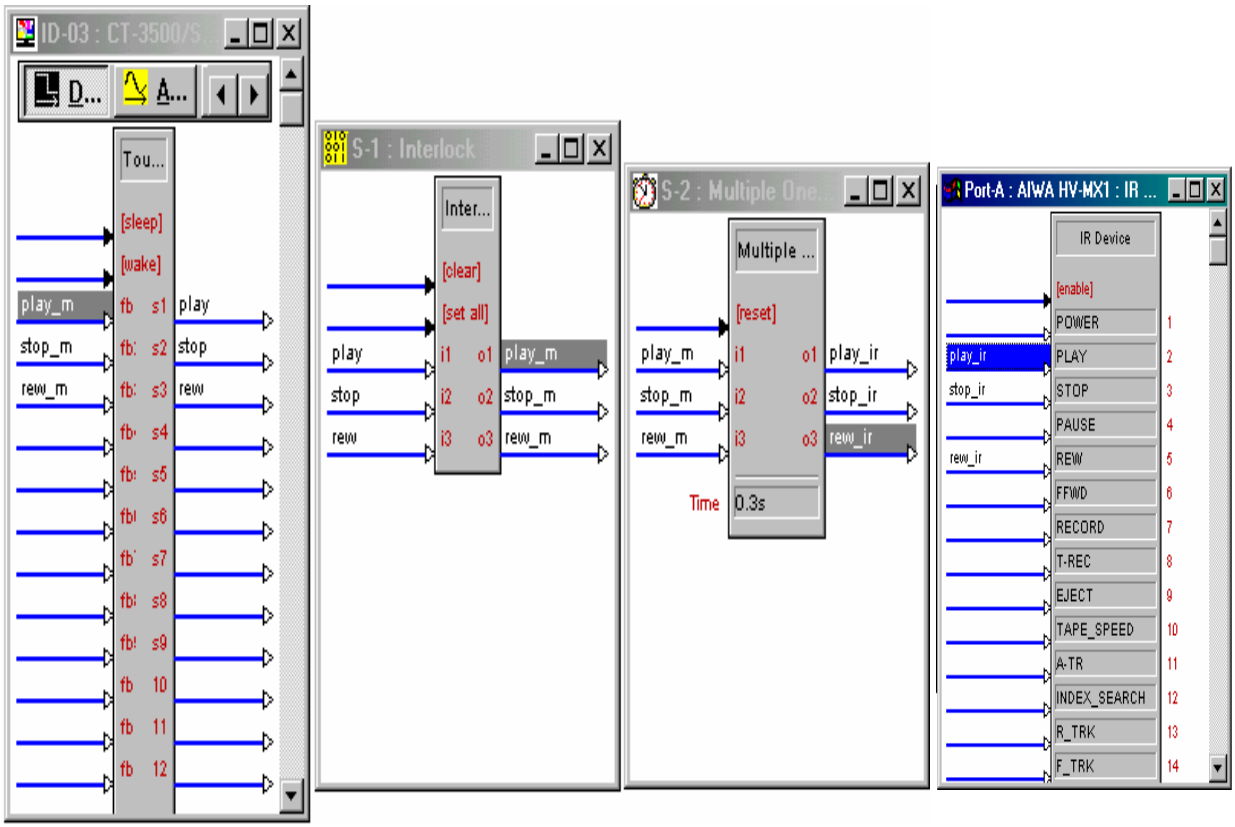
Insert the first logical symbol

Functionality of in and outputs – with and without ( )

F1 – where to find info on symbols

Used Logic Symbol(s): TOGGLE

Example 3



More logic

Insert more symbols to illustrate feedback and signal Pulsing: Interlock is used for the feedback, the MOS (MMV) is used to pulse the signal

IR does not produce any feedback – so you need to do it via SW –

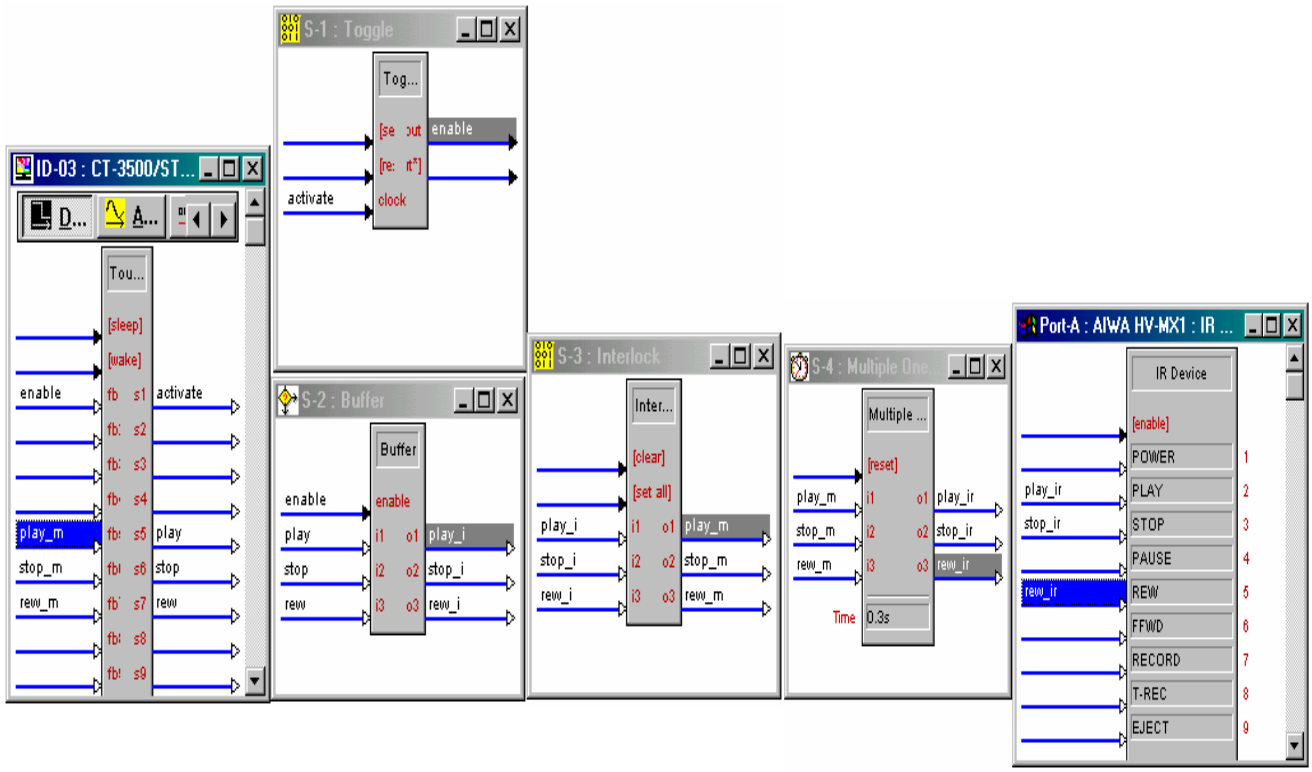
Use an IR driver instead of the relays: where to find it in the database on the configuration page.

This is a logical example and it does not mean that you have to do it this way. Ex 1 – direct control – can be perfectly used to work with IR.

" **Speedkey Name**": insert symbols by typing the speedkey name, you do not have to drag and drop (faster).

Used Logic Symbol(s): INTERLOCK, MOS

Example 4

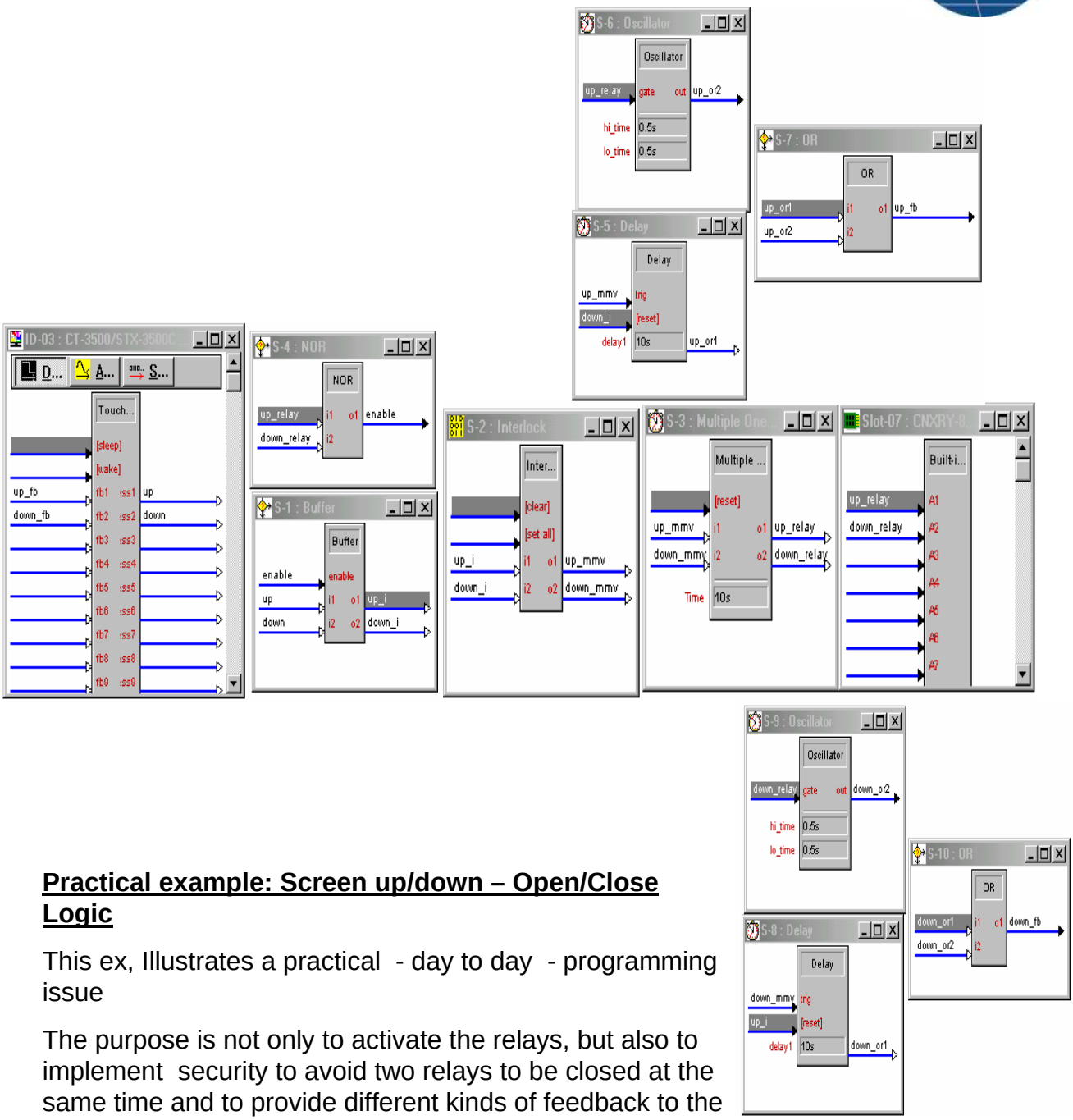


More logic: Blocking signals

Use ex.3 as startpoint and add a BUFFER symbol

Used Logic Symbol(s): TOGGLE, INTERLOCK, BUFFER, MOS

Example 5



Practical example: Screen up/down – Open/Close Logic

This ex, Illustrates a practical - day to day - programming issue

The purpose is not only to activate the relays, but also to implement security to avoid two relays to be closed at the same time and to provide different kinds of feedback to the button: move up/down and position up/down

Used Logic Symbol(s): BUFFER, NOR, MOS, INTERLOCK, OR, DELAY, OSC

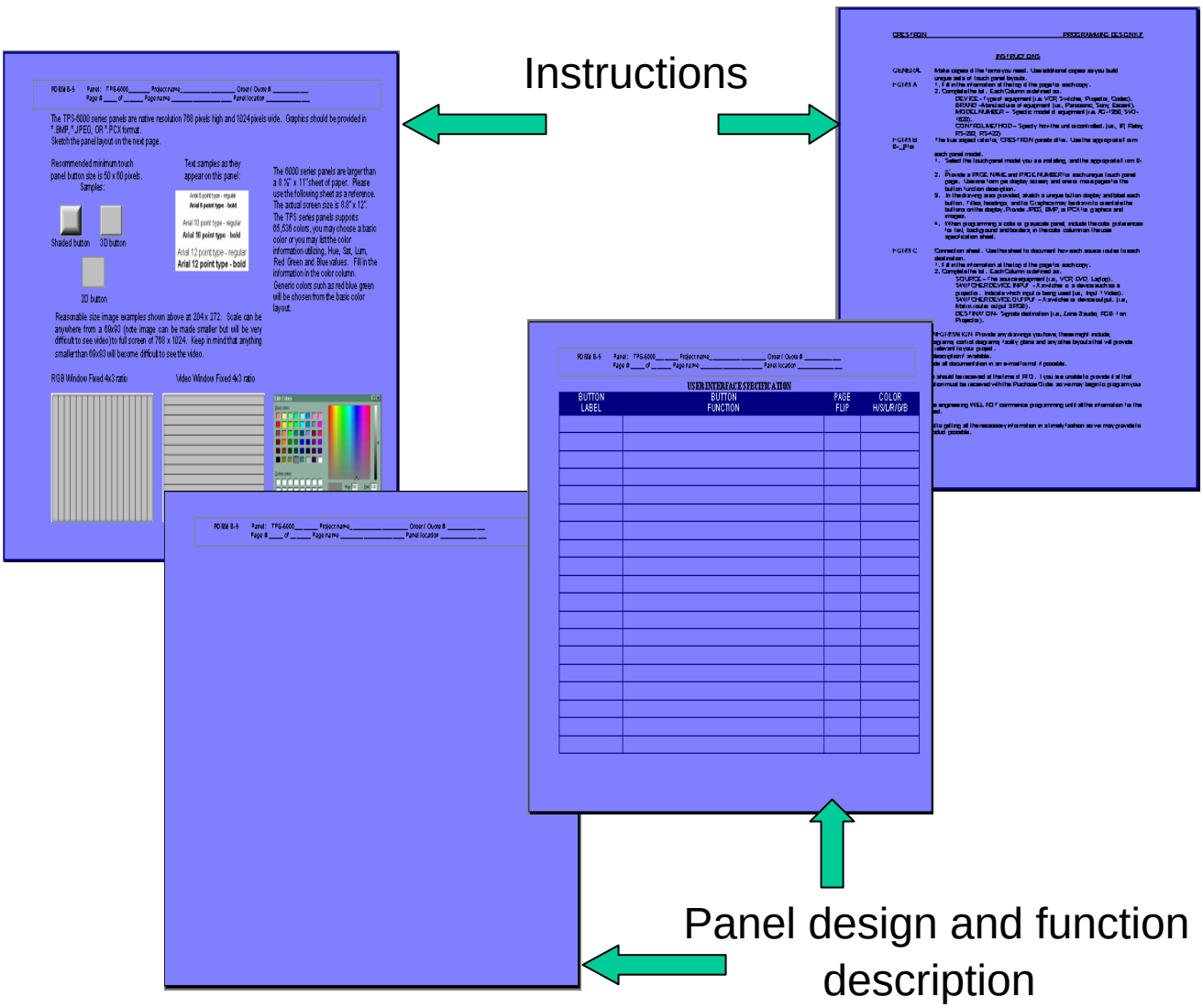
7. Touchpanel design with VTPRO-e and layout activation with Simpl Windows

1) Make a script with the PDK
2) Make screen per screen and explain all the VTPRO-e features when you go along
3) Upload the result in the TP and do the first test (page flip)

1) Make a new SW program
2) Activate items on the pages one by one. Each time you finished one, let them upload and test this item.
3) Build up a clear structure (subsystems,…)

Create a script with the “program design kit”.

It is very important to use it.This is the only standard to start a program efficiently



First Page:

Welcome screen

Insert: Image, text window and a clock

Add a Transparent button with **"Page Flip"** function

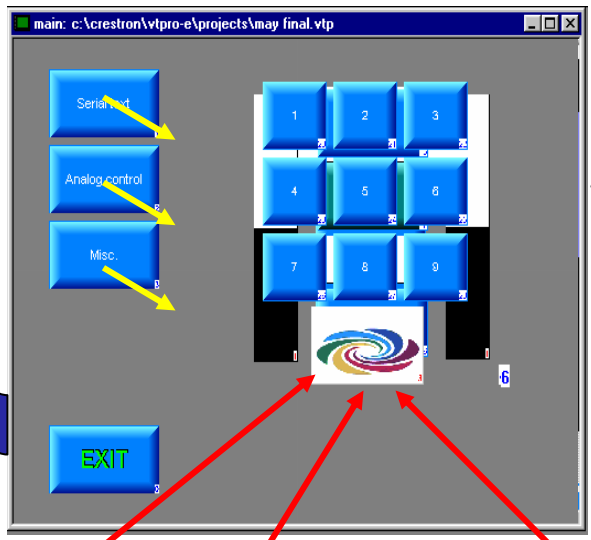
Second Page:

Working screen

Add 3 buttons for subpages

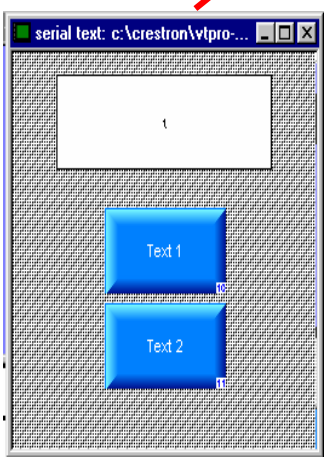
Add **EXIT** button with**"Page Flip"** function

Create and add **subpages** to this screen.

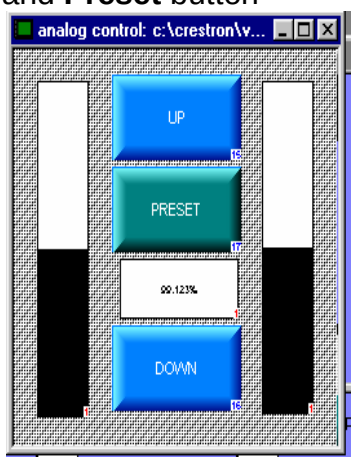


Subpages: Create 3 identical subpages.

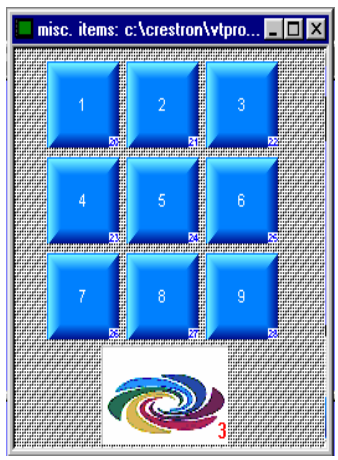
Serial Indirect text: border with text field and 2 buttons



Analog: slider, gauge, %. Up, Down and **Preset** button

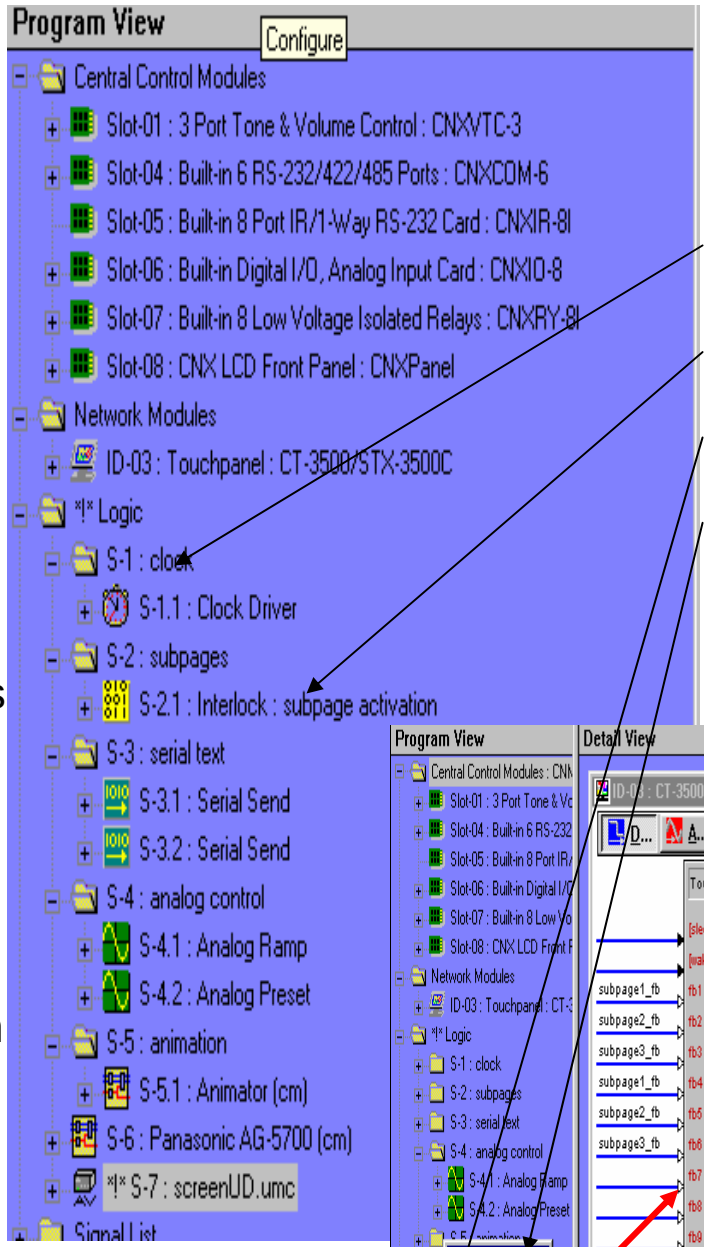


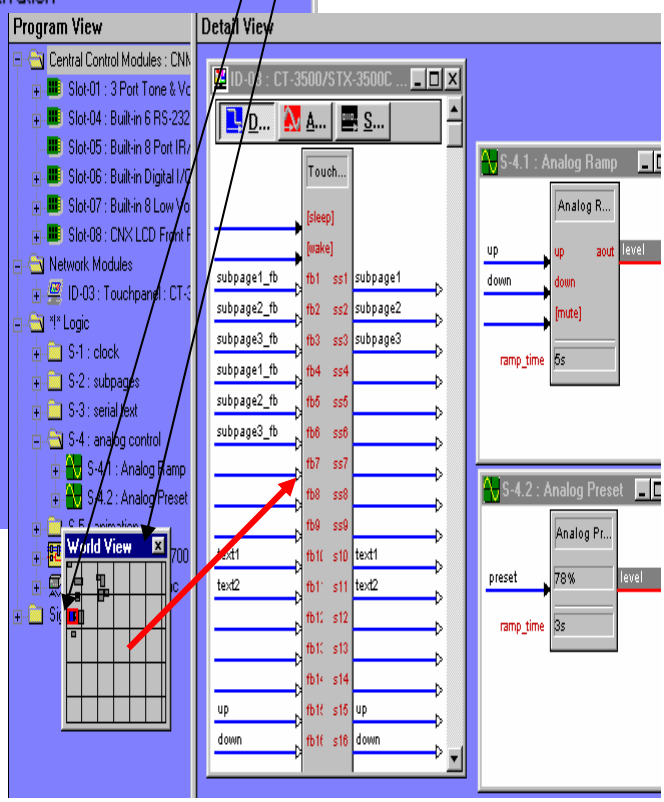
Misc: 9 buttons for **macro control and serial send/ receive + animation**



productspluspeople

SIMPL Windows Program Activation





Program structure:

- **Subsystems**
- **Comment symbols**
- **Bookmarks**
- **Worldview**

Subjects:

A. Clock

B. Subpages

C. Serial Indirect Text

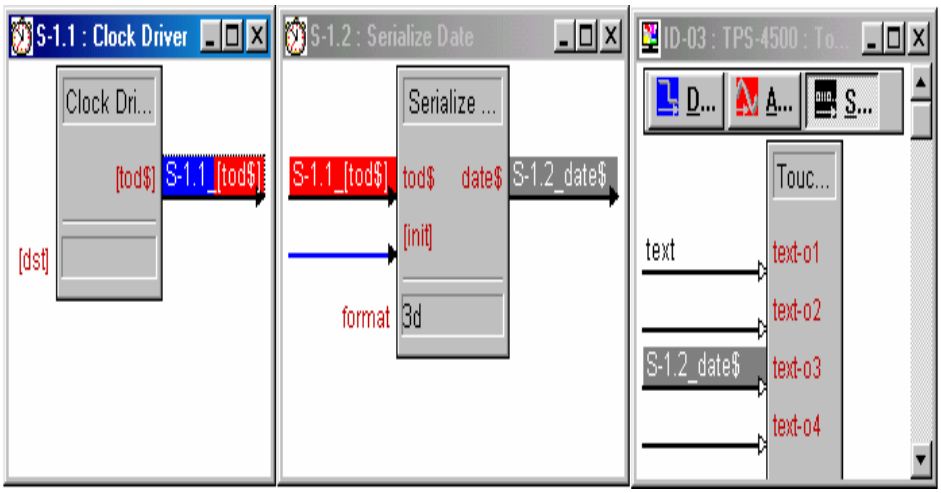
D. Analog Control

E. Animation

F. Modules

G. Serial

A. Clock/date activation



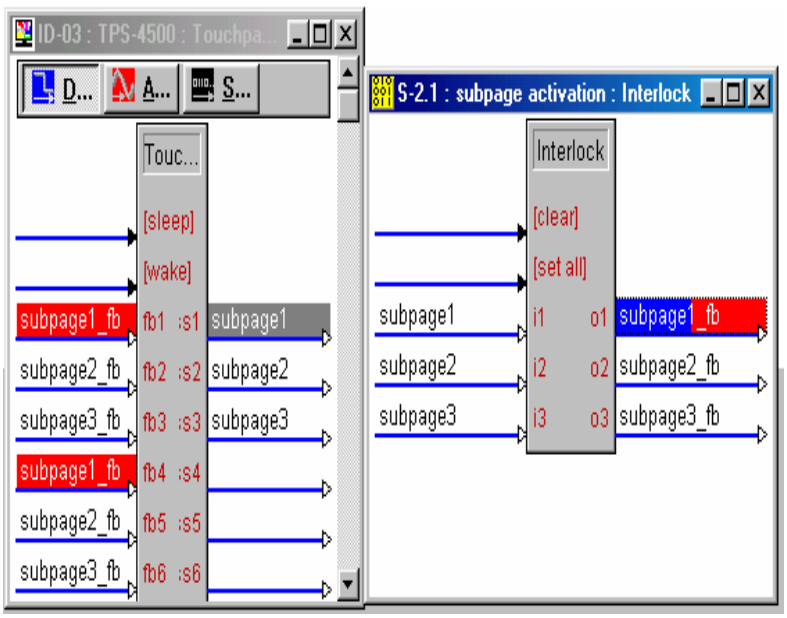
- DST format **4** for Europe (only with 2-series!, otherwise leave open)

- Serialize date (7 formats) sends out text string to be displayed in a normal text field

-**Symbols:** Clock driver, Serialize Date (Date$)

- The date “Text” string is only send at program startup or at midnight, update possibility via the [INIT] function on the date symbol

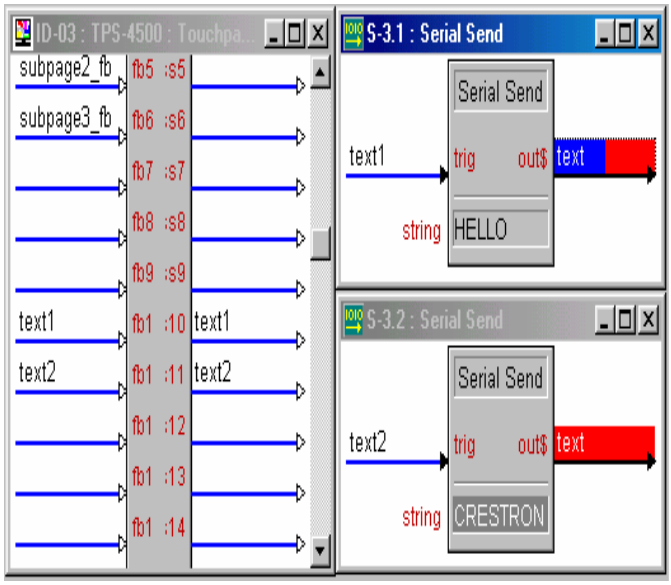
B. Subpages

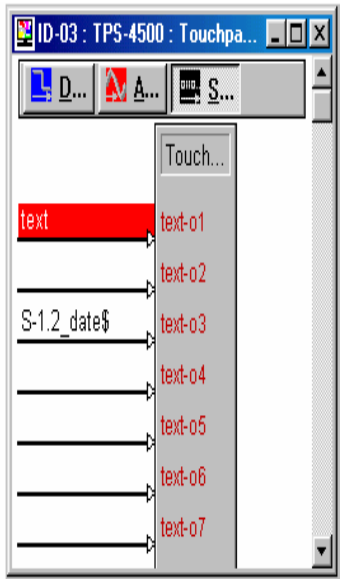


Subpages (4,5,6) are made visible by a high digital signal. In this case the Interlock symbol provides this high signal and so keeps the “feedback” high.

(*)Full pages can also be given join numbers, but in the contrary to subpages they just need a digital pulse to be made visible and stay visible = alternative page flip!

C. Serial Indirect Text





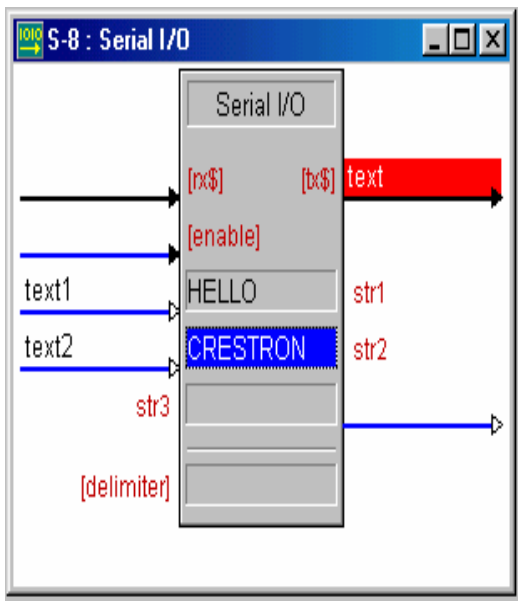
Direct feedback: only possibility

Serial signals: mulitiple sources can go to one destination. Exception only valid for **serial** and **analog** signals.

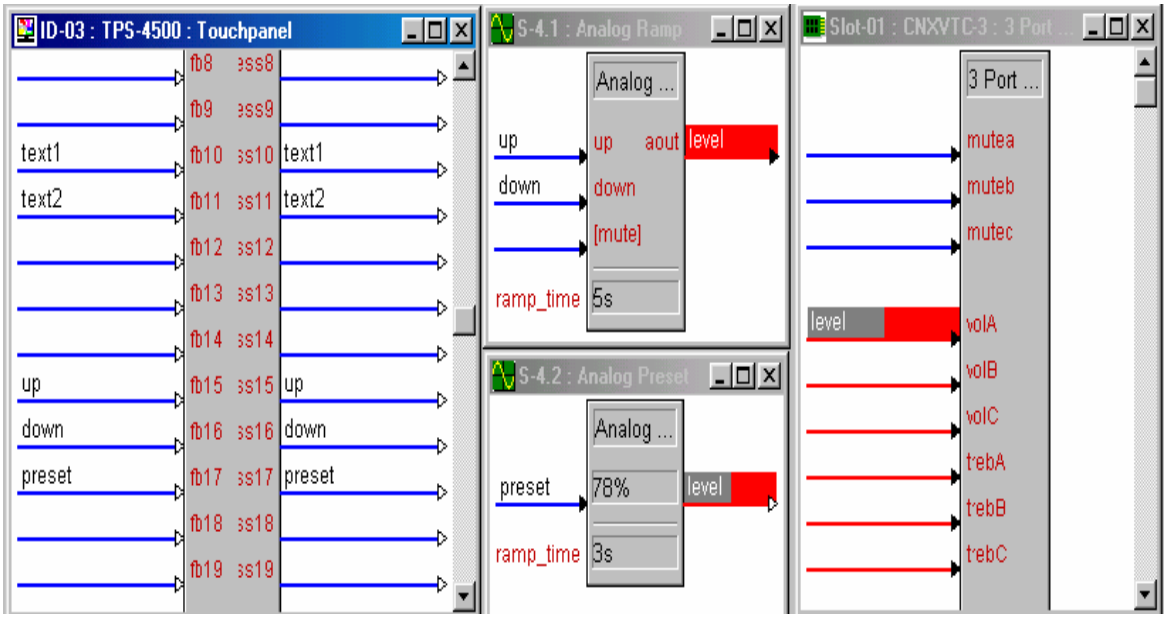
Text string: 80 characters max. If you want to have the text displayed on multiple lines then you need to use “\x0D”.

Example: Hello**\x0D**Crestron

Alternative: use a Serial I/O: you can have multiple text messages/strings in 1 symbol



D. Analog control



Analog signals: correspond with analog channels that work with a value between 0 and 100% (0-65535 or 0000h-FFFFh) – bi-directional

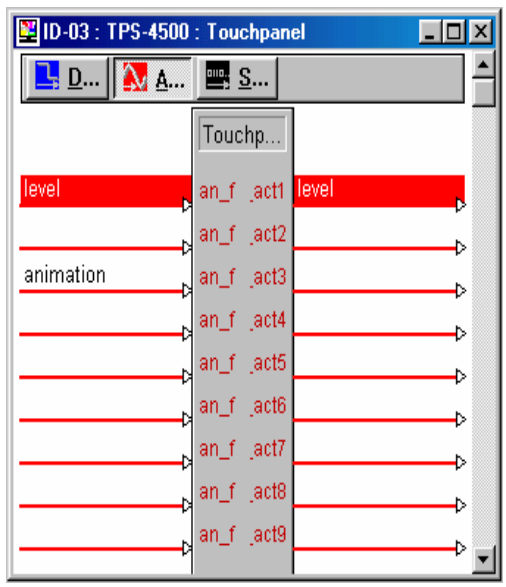
Basic Analog Symbols: RAMP, PRESET, INIT

Analog signals: mulitiple sources can go to one destination. Exception only valid for **serial** and **analog** signals.

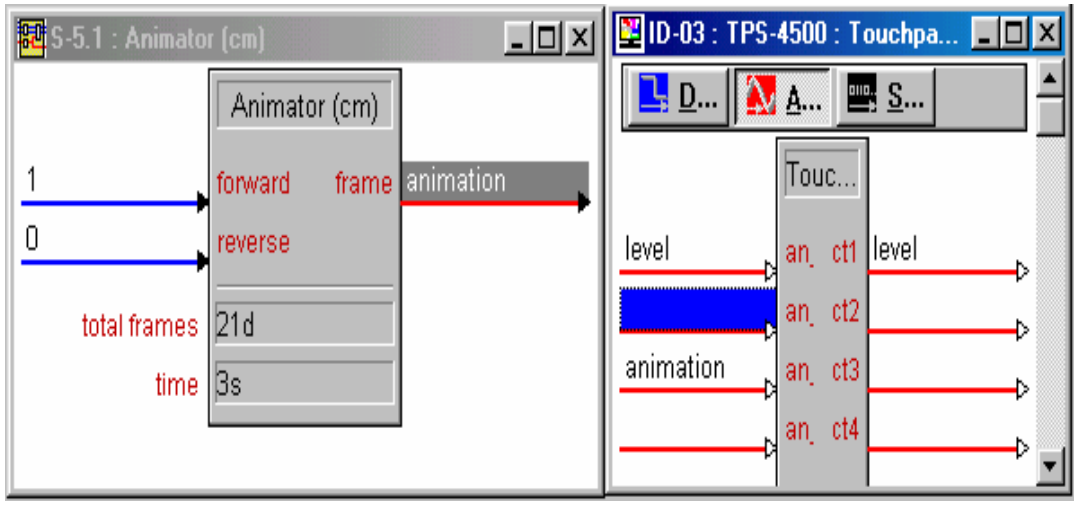
Gauge and % display: analog signal comes in at the feedback side.

Slider: analog activity and feedback signal names need to be identical to make the slider work. (*)

Direct feedback buttons: only possibility



E. Module implementation: Animator



Crestron Modules: (p64 of the SW manual CDRom)

Crestron Macros are prepackaged logic programs. A Crestron macro is a set of pre-written and debugged logic used for controlling a particular device or performing a function. The use of macros saves programming and debugging time since a large portion of the symbol – signal functionality already exists inside the macro.

Module implementation rules:

All inputs, outputs and parameter fileds need to be filled in, even if you do not need all of the functions for your program, otherwise:

-Compilation errors: Signals without driving -source, signals with destination (!)

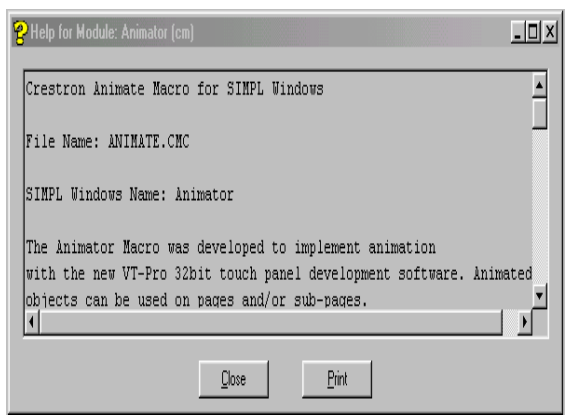
-Malfuntioning of the macro

-Non functioning of the macro

Solution: use DUMMY signal names – explain how your can avoid the error messages and what the disadvantages are from using the “0”

For more info Press F1 (some older modules do not have a help File

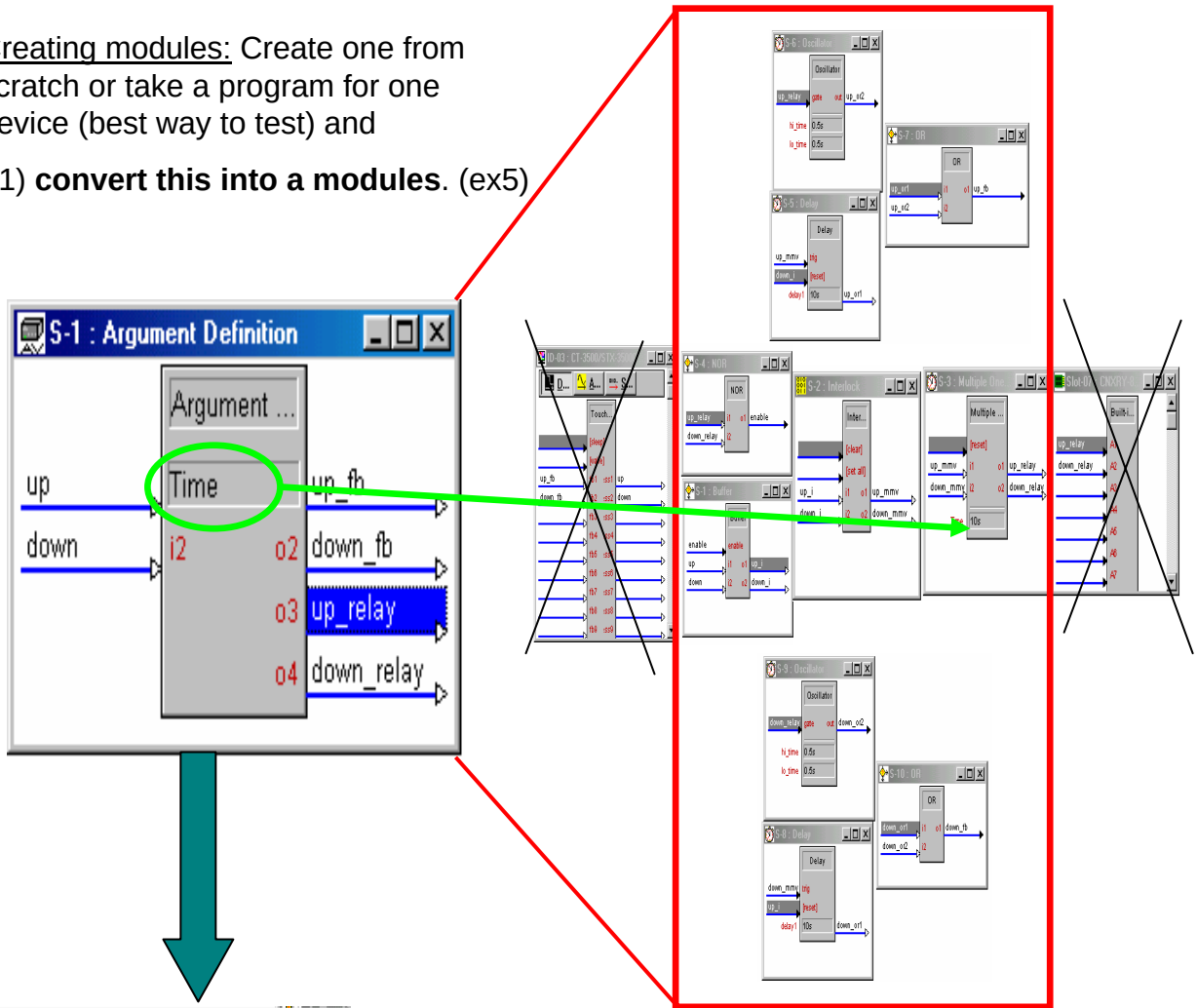
F1

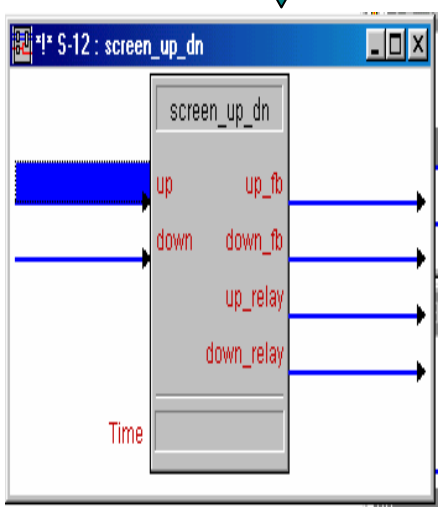


F. Creating Modules

Creating modules: Create one from scratch or take a program for one device (best way to test) and

(1) **convert this into a modules**. (ex5)





(2) Hardware is stripped of

(3) DEFARGS (Define Arguments) symbol is added: to define inputs, outputs and parameters

(4) Parameter: can be made variable: ex. DEFARGS: “Time” and the parameterfield of the MOS “#Time”. (Make sure to fill in the DEFARGS first!)

(5) Create a “HELP” file: Select “project” and “Program Header”

(6) Save as a .UMC file in the User Macro database.

G. Serial communication

RS-232, RS-422, and RS-485 are all physical standards for serial communication:

- The **bi-directional** data communication has the advantage that it can produce "**life feedback**"

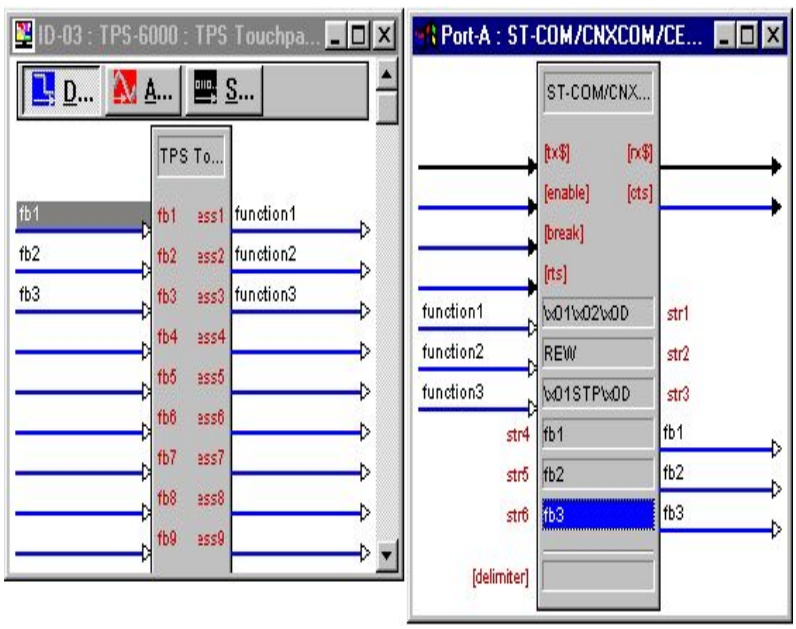
- The **data format, or protocol**, that a controlled device is expecting will be described in the unit's manual – it is different for every manufacturer and sometimes even every model

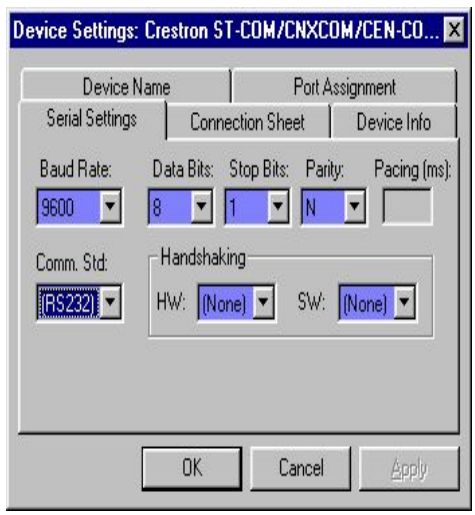
- **Serials settings**: Depending on the controlled device the data will have to be send out following a certain way, more in particular the speed at which it communicates (baud rate), the error checking (parity), the number of data bits and the number of stop bits. In addition, a given device may require hardware (RTS/CTS) or software (XON/XOFF) handshaking, which controls the flow of data between two devices

-**Cables:** every serial controlled device has to be connected to a CRESTRON system with a non-standard cable. This cable differs from unit to unit. The regular updated **CABLE DATABASE** contains a lot of cable diagrams.

Manual programming

Comport fields accept the protocol in Hexadecimal, ASCII or a combination of the two





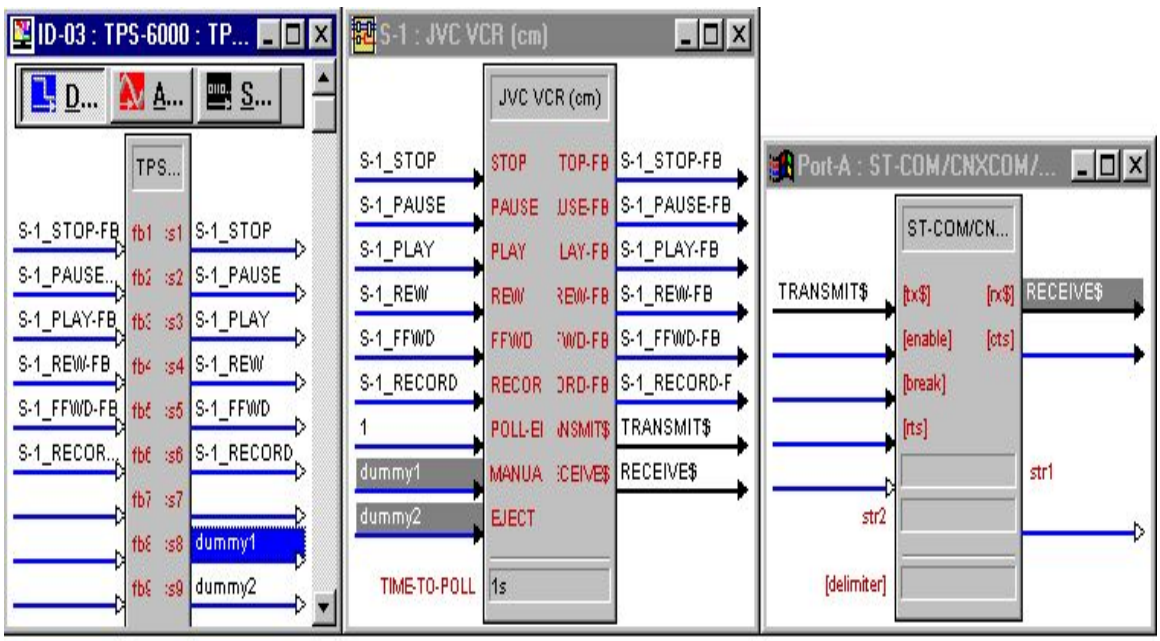
Serial setting are to be set in the **configuration screen**

G. Serial communication (2)

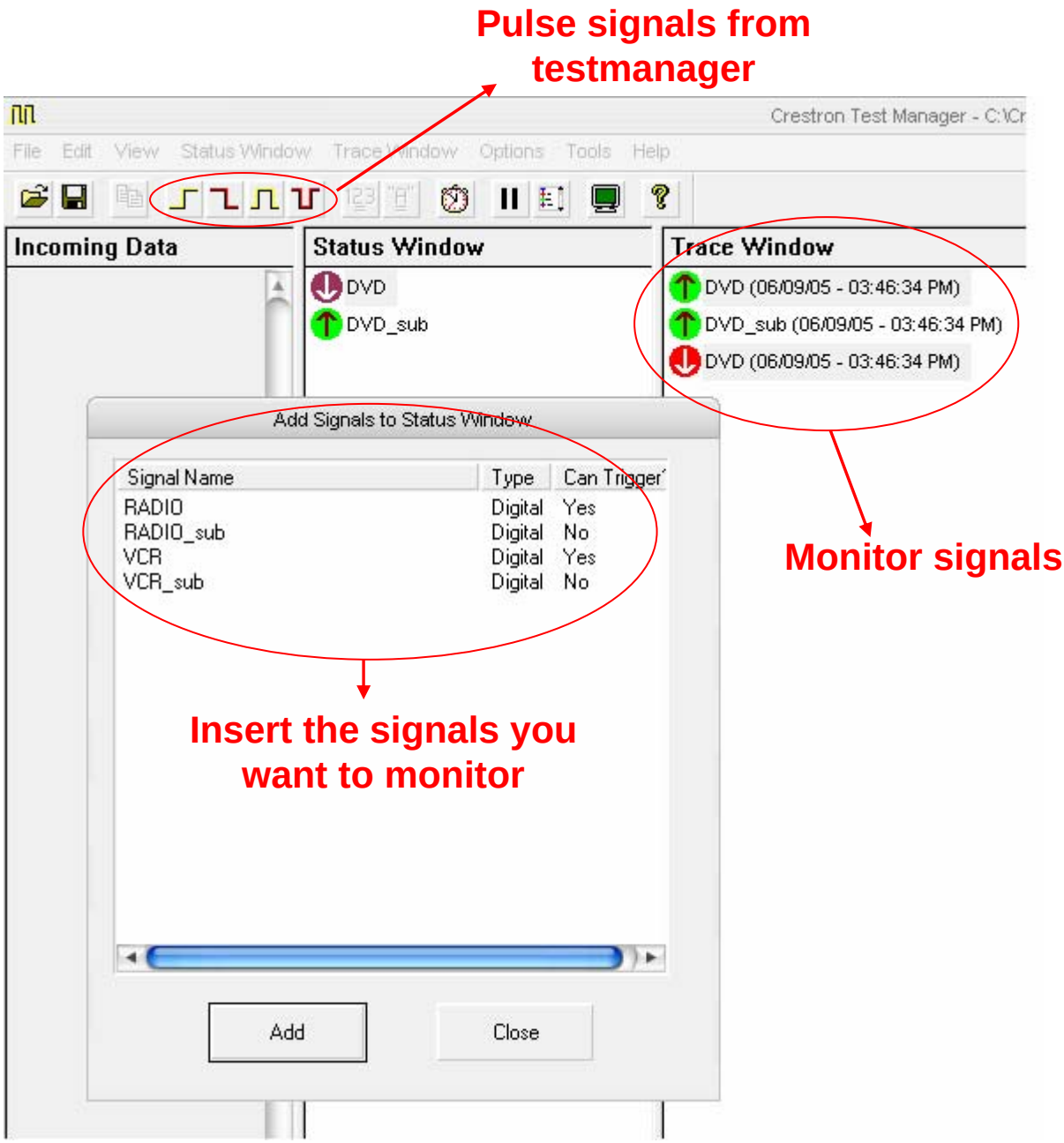
Using a serial module:

Crestron offers logic modules (also known as macros) that have been written for many devices. Modules are self-contained SIMPL programs that look like symbols and can be dropped into a larger program to generate all the proper control codes automatically.

Follow the “Module Implementation Rules”

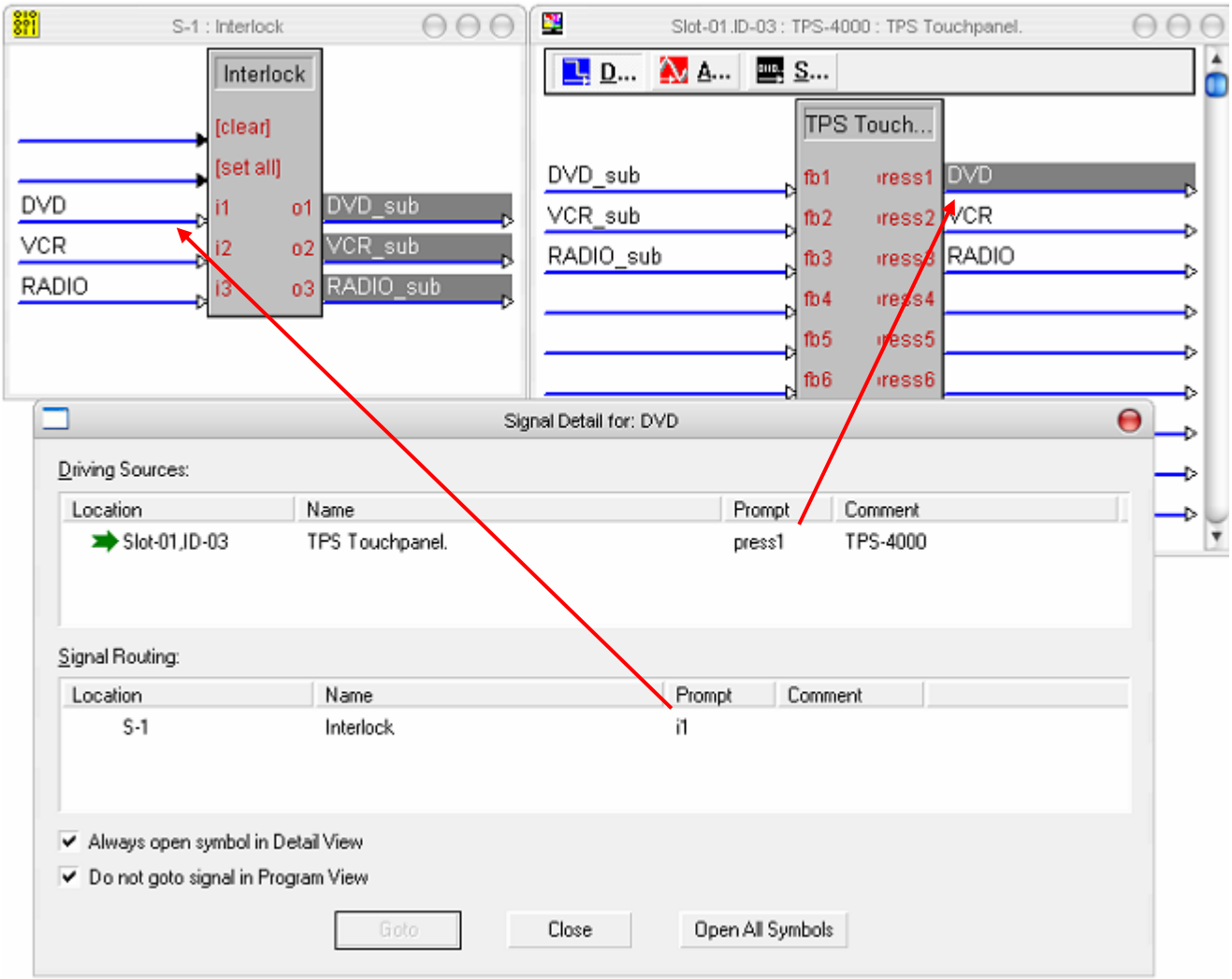


H. Test manager:



I. Signal routing

Use the F2 key to find out the signal routing

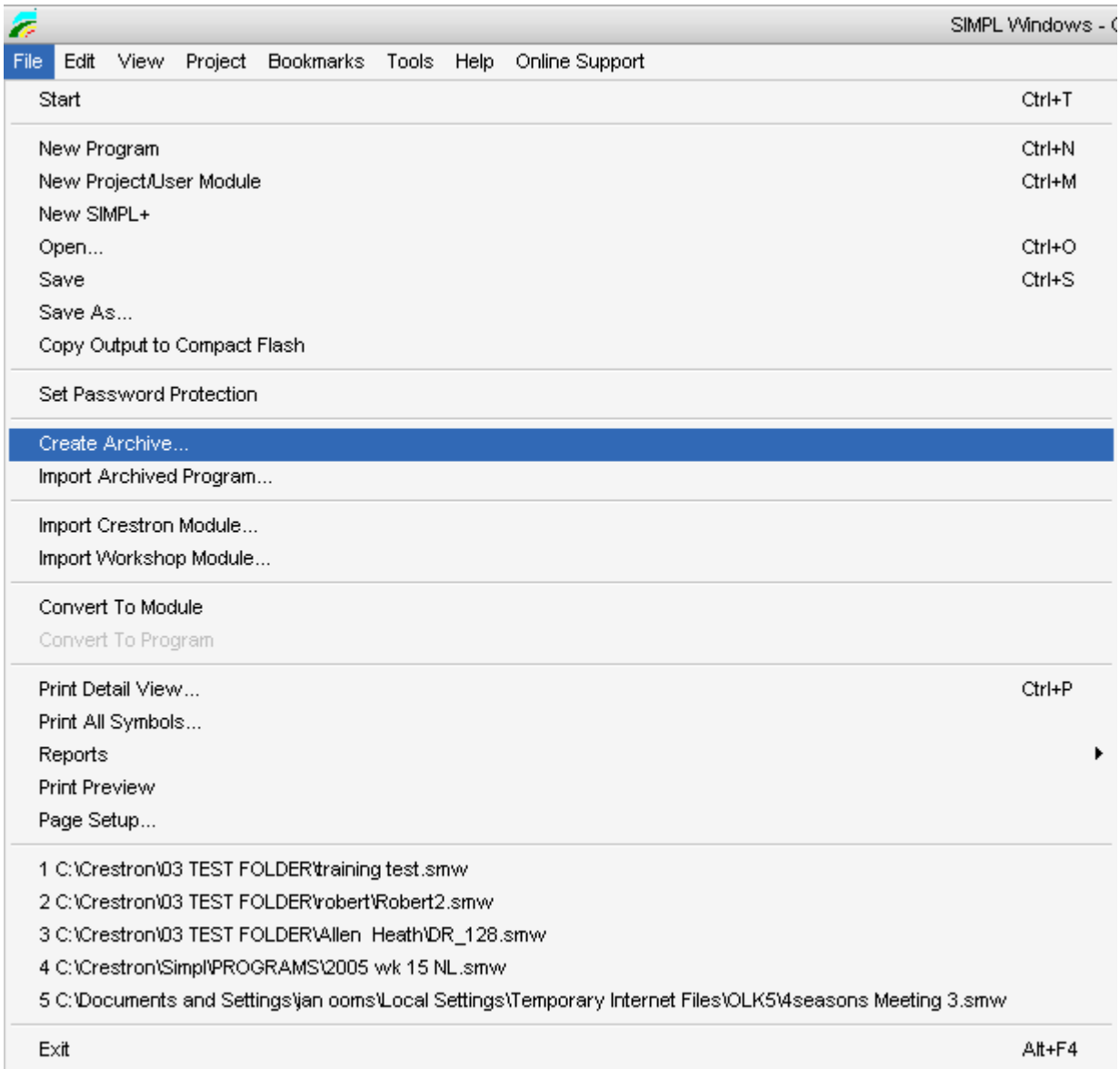


Signal and symbol will automatically be opened and highlighted in detail view when you dubbelclick on the signalname

Use the F3 key to highlight signals with the same name

Appendix

What's the best way to save a program?

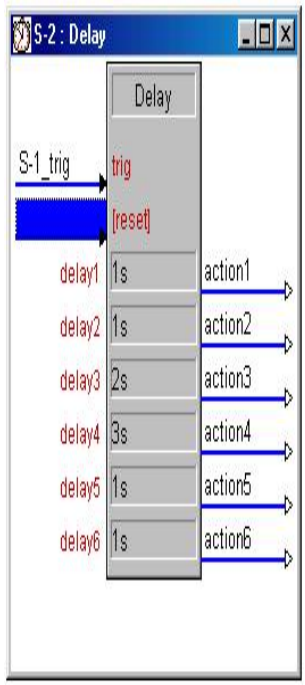


An 'archive' is a .zip file that contains all files used by the program (.smw, .umc, .usp, .ir).
This means it will also save used macro's and IR drivers (very usefull if you need to send your program to tech support)
You can open the .zip file by using 'import archived program'. This will make sure all used drivers are placed in the correct folders.

Sequences

It is possible to trigger multiple actions with only one button press. Actions like this are used to start up or shut down a complete room/installation: lights go to the right level, projector switches goes on, screen comes down, switcher changes to the right input/output,… all activated by only one button press.

Useful Symbols:



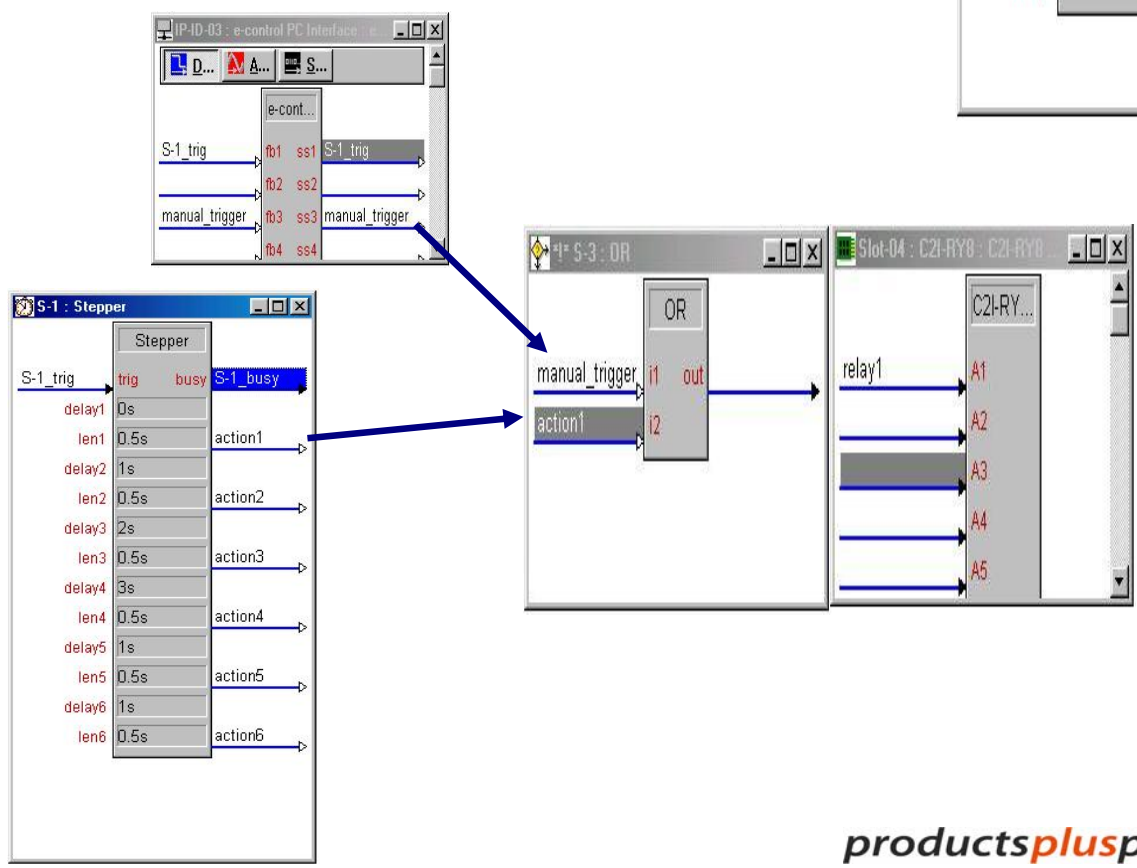
The Stepper symbol drives its output signals high on the rising edge of <**trig**> after the corresponding <**delay**> expires. Each output then remains high for the period specified by its corresponding <**len**> parameter. Any subsequent changes in <**trig**> have no effect until all outputs are low again.

The <**busy**> output goes high if any outputs are high, and low when all outputs are low.

The Delay symbol drives each output to the level of the <**trig**> input after the corresponding <**delay**> expires. Note that all specified delays are independent of one another; that is, there is no cumulative delay effect.

The optional <**reset**> immediately drives all outputs to the level of <**trig**> (with no delay) for as long as <**reset**> is high.

By using an **OR symbol** you can trigger a function with multiple driving sources



Analog Values and the INIT symbol

Analog Initialize

Speed Key Name : init

Signals/Parameters

Single Input Form

•One digital input: <**trig1**>

•Any number of analog outputs: <**aout1**> through <**aout*N***>

•For each output, one corresponding parameter: <**value1**> through <**value*N***> (*See* Numeric Formats)

Single Output Form

•Any number of digital inputs: <**trig1**> through <**trig*N***>

•One analog output: <**aout1**>

•For each input, one single-precision parameter: <**value1**> through <**value*N***> (*See* Numeric Formats)

Description

In the single input form the Analog Initialize symbol drives each output to the value specified by its corresponding <**value**> parameter, with each rising edge of the input signal.

In the single output form the symbol initializes the value of the output on the rising edge of any of its inputs. The output will be set to the <**value**> parameter that corresponds to the input that last goes high.

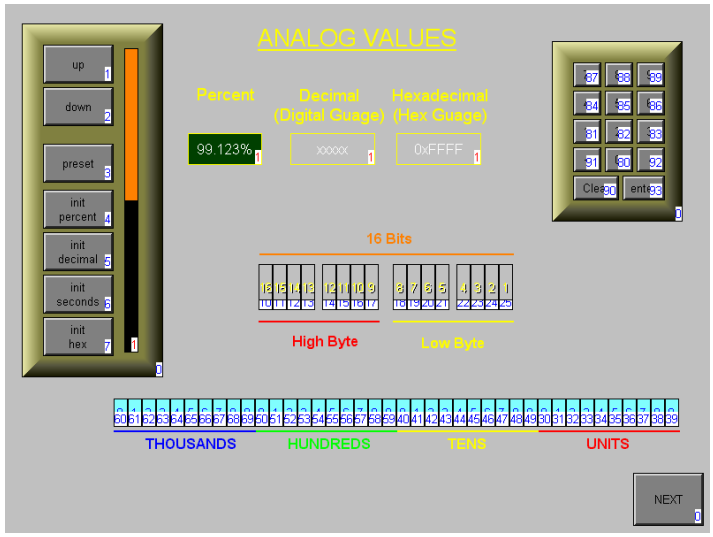
At startup all outputs have a value of 0, except in the single input form when the input is given the signal name 1. In this case the outputs will have the value specified by their corresponding <**value**> parameters.

Conversion tables for Hex, Binary and ASCII can be found in the SIMPL Windows Help File

Numeric Values

Numeric values can be expressed in a number of formats, where the character in parentheses represents the format identifier:

- (d)ecimal
- (h)exadecimal
- (%) percentage
- (s)econds
- (t)icks (1 tick = 1/112.5 seconds)
- (')character(') (single byte)



The allowable range of analog values expressed in each format is as follows:

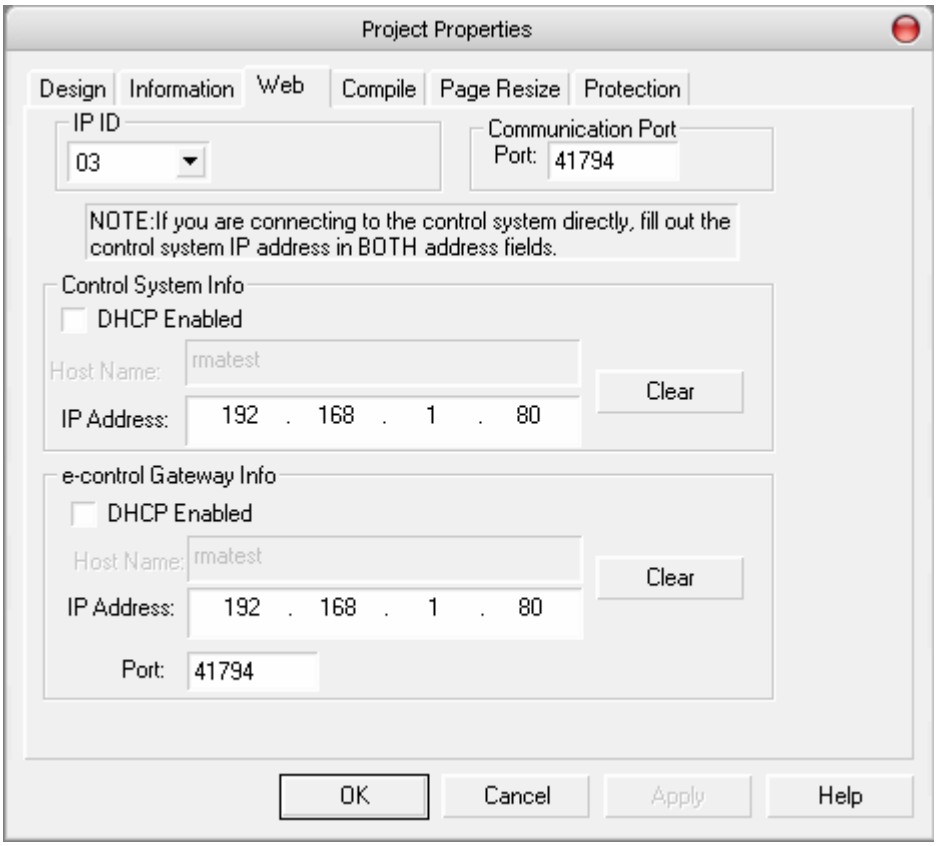
Format	Minimum	Maximum
Decimal	0d	65535d
Hexadecimal	0h	FFFFh
Percentage*	0%	100%
Seconds**	0s	582.53s
Ticks	0t	65535t
Byte	' ' (space, ASCII 20h)	'~' (tilde, ASCII 7Eh)

*Percentage and seconds formats can be expressed with precision of .01% or .01s.

**Double precision time values range from 0.0 seconds to 19,088,743 seconds.

Every parameter has a default format if none is specified when the symbol is defined.

Creating an Xpanel project



Step 1. Create a new project and select Xpanel as panel type.

Step 2. Go to the project properties (web tab) and fill in the required IP adresses (gateway is in most cases the same as the IP adress of the control system) and select an IPID.

Step 3. Go to SIMPL windows and add an Xpanel on the ethernet. Make sure it has the same IPID as in VTPRO.

Step 4. Fill in 127.0.0.1 as the default adress in the properties of this Xpanel

Step 5. Upload the webpages and program to the control system

Step 6. Open explorer and fill in the IPadress of the control system.

Assignment