ISTITUTO
DI TECNOLOGIE DELLA
COMUNICAZIONE,
DELL'INFORMAZIONE
E DELLA
PERCEZIONE

Scuola Superiore
Sant'Anna

Course Introduction to Matlab and Simulink Simulink/2

Emanuele Ruffaldi

May 18th, 2017

http://www.eruffaldi.com/wp/introduction-to-matlab-and-simulink/

Scuola Superiore Sant'Anna, Pisa

Recall first Lecture and Web Tutorials

- Create Simple Model (Link)
- Model-based Design (Link)

Exercise on Expressing Differential Equation

- Take the basic examples from ode45 help page and realize them in Simulink
- Check in particular vdp

Sample Time

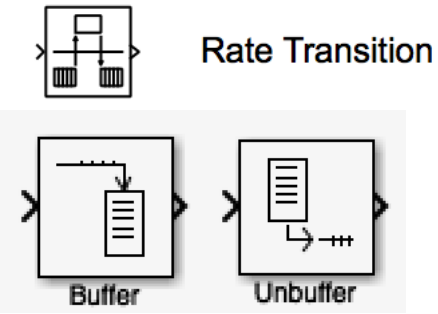
- Sample time regulates the execution of each block and when the data is exchanged
- We have several types of sample types (Web) expressed via a pair of numbers [major,minor]
  - Inherit [-1,0] or -1
  - Constant [Inf,0] or Inf
  - Continuous [0,0] or 0
  - Discrete Tk
  - Triggered (specify via -1)
  - Asynchronous (e.g. interrupt)

Multi-rate Sample Time

- In complex scenarios there are various discrete sample times. These have to be multiple of the base simulation time

$$0 \leq n \leq \frac{T_{sim}}{T_s}$$

- Data between rates can be exchanged in two ways
  - Data Transitions
  - Data Buffering
- Simulink can automatically insert Rate Transitions



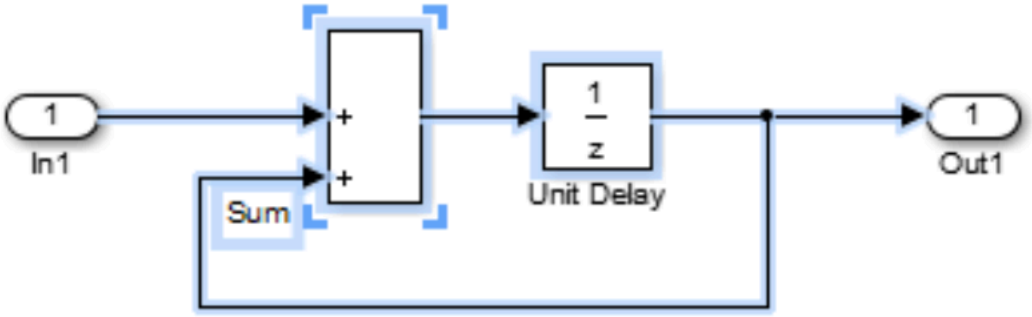
Dealing with Complexity

There are three constructs that can be used for dealing with complexity

1. Subsystems
2. Bus
3. Enumerations

Subsystems

- Subsystems allows for organizing the Model in parts, also providing functionalities
- Regular Subsystems can be created from the Library Browser or from a selection of components via Context Menu
  - Subsystem Expansion is the opposite operation



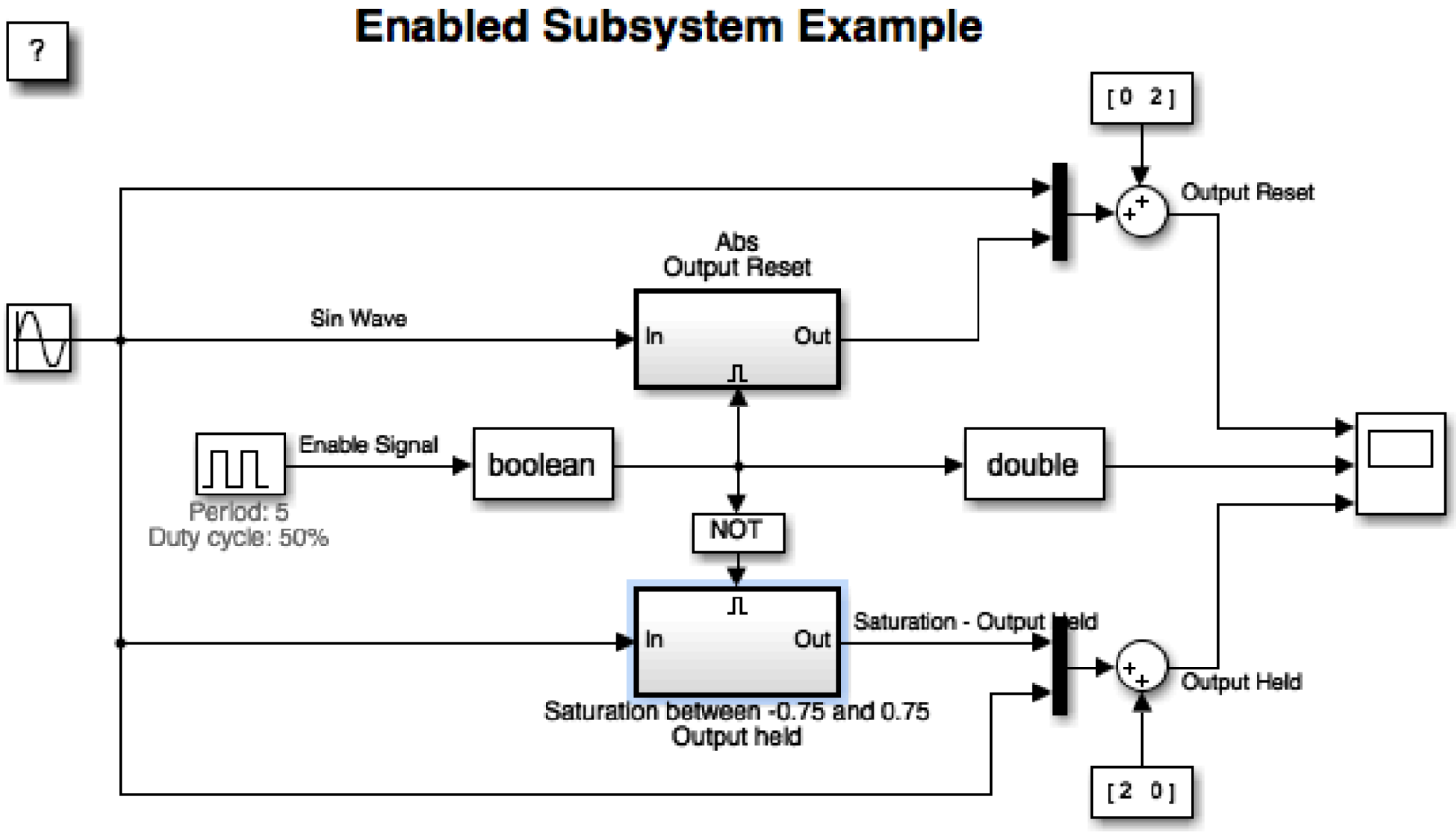
Regular Subsystems

- Regular Subsystems are, by Default, virtual in the sense that they are equivalent to the separate components
- Subsystems are connected to the containing System via Ports
  - Input and Output Ports
  - Each Labeled
- Subsystems can be navigated
  - Same Window: double click
  - Another Window: CTRL + double click
  - Another Tab: SHIFT + double click
- Try navigating an example complex system: **sldemo_enginewc**

Enabled and Triggered Subsystems

- An enabled subsystem is controlled by an external port that marks the activity of the subsystem
  - Reset subsystem: outputs reset when disabled, internal state reset on enable
  - Held subsystem: everything is kept
- A triggered subsystem is active only in the rising/descending value of the trigger port

Enabled Subsystem Example

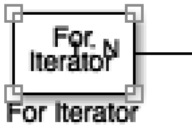


>> enablesub

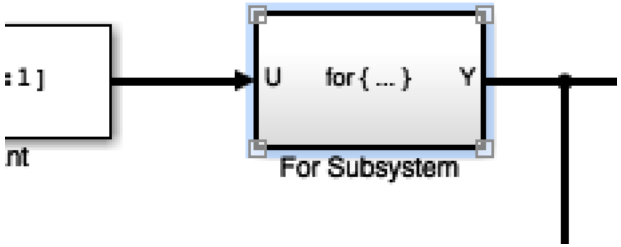
Control Flow

- Some systems cannot be represented by equations only but need to use some control-flow logics much like programming languages
  - Conditionals (If)
  - Numbered Iterations (for)
  - Conditioned Iterations (while)
  - Alternatives (switch)

For Subsystem



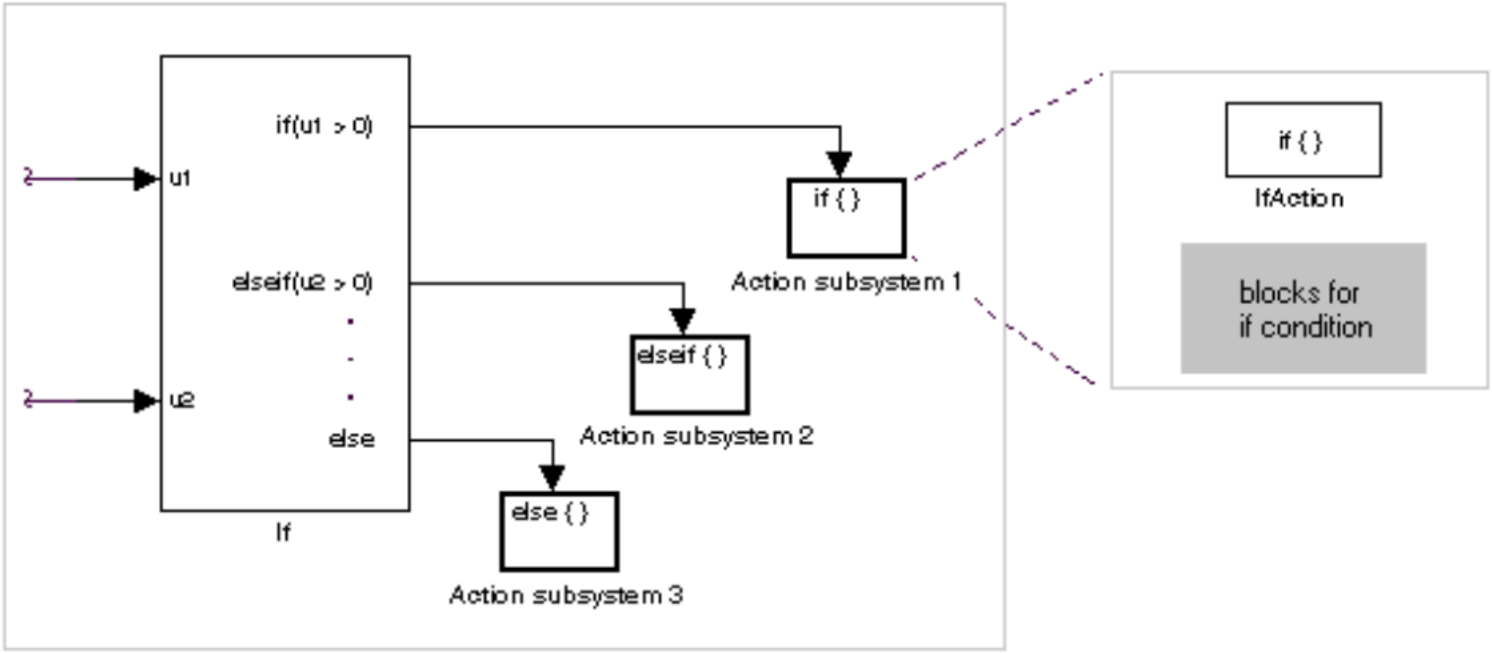
- Repetition of a Subsystem a number of times depending on parameter or external condition
- Iteration count can be fixed or port



- Sample time of the System is **Triggered**

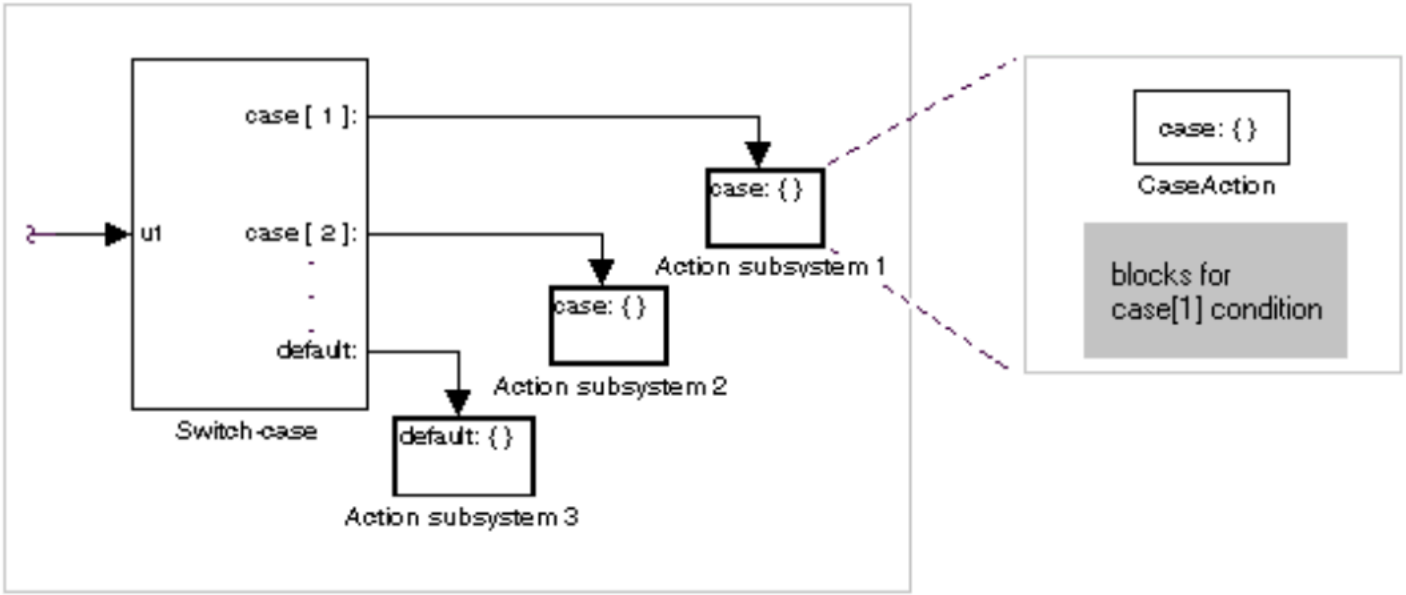
If Block

- The if block emits one action for every condition and the Action needs to be handled by an Action port of a corresponding Subsystem



Case Block

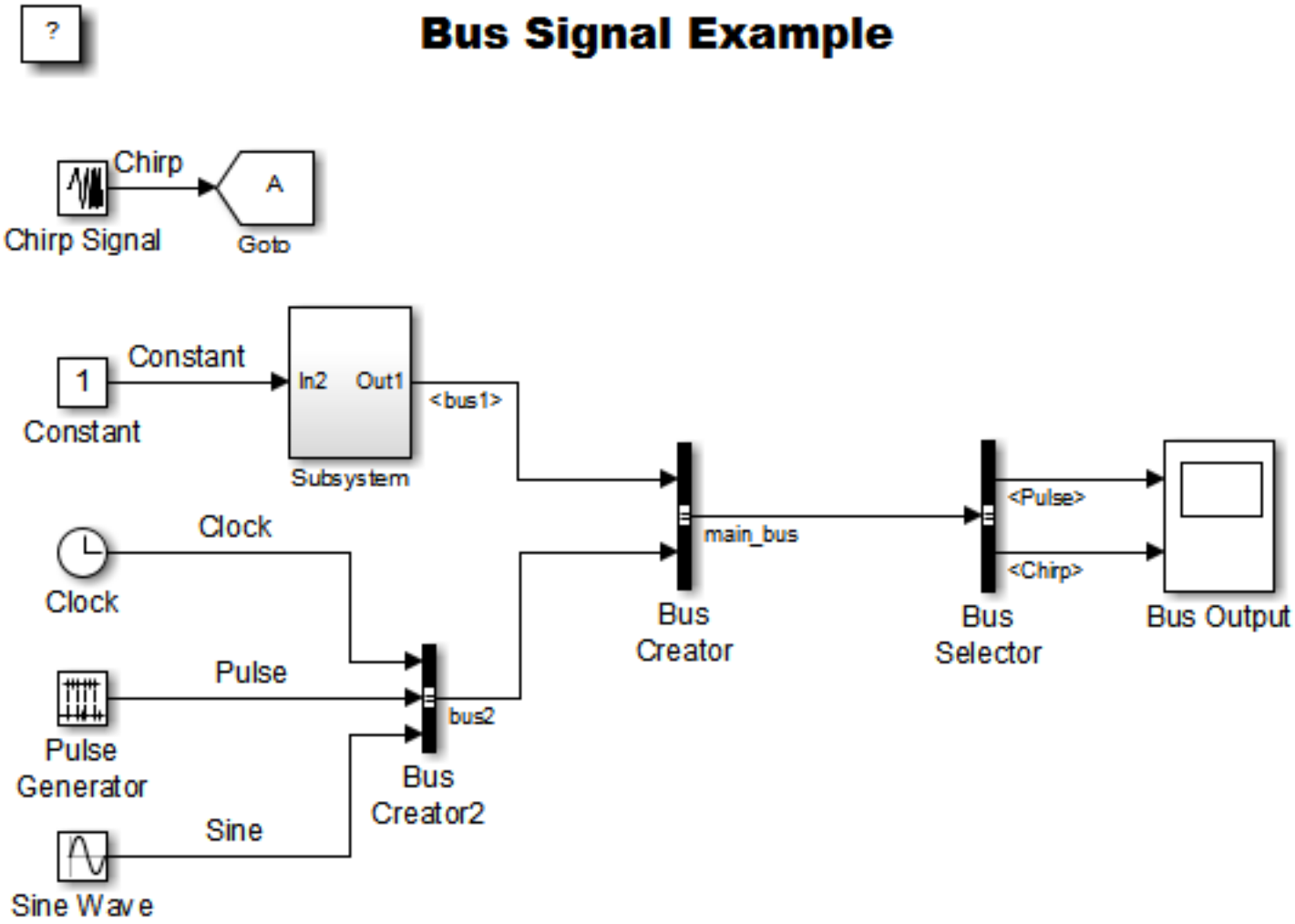
- Similarly to the If Block the cases conditions are defined in the Case block and they trigger Actions



Bus

- So fare we have exanimated signals containing data with a single content, even if in matrix form
- For the sake of organization it is better to structure signals in groups called Bus that correspond to the struct/record of programming languages
- References

Bus Example



Enumerations

- For better Simulation organization it is worth using enumerated types
- An enumeration is a subset of the integer space for which names are associated to numbers
- This corresponds to enum in C/C++
- Enumerated types allows to better clarify the state of the Simulation also in the case of Case blocks

Defining Enumerations

- Enumerations are special types of Matlab classes derived from Simulink.IntEnumType
- They can be defined using a classdef statement or via a single function invocation
- The class definition needs to be loaded into the Model Workspace

```
classdef ColorChannels < Simulink.IntEnumType
        enumeration
                Red(0)
                Green(1)
                Blue(2)
        end
end
```

```
Simulink.defineIntEnumType('ColorChanne
ls', ...      {'Red', 'Green', 'Blue'}, [0;1;2],...
'Description', 'Basic colors', ...
'DefaultValue', 'Red', ...
'AddClassNameToEnumNames', true);
```