Simulink

05 February 2021 12:46 PM

1) What is Solvers in Simulink ?
   - To simulate a dynamic system, we compute its states at successive time steps over a specified time span. This computation uses information provided by a model of the system
   - The process of computing the states of a model in this manner is known as solving the model. No single method of solving a model applies to all systems.
   - Simulink provides a set of programs called solvers. Each solver provide a particular approach to solving a model.
   - A solver applies a numerical method to solve the set of ordinary differential equations that represent the model. Through this computation, it determines the time of the next simulation step
   - Simulink provide Two main types of solvers:
     - Fixed Step Solver :-
       Fixed-step solvers, as the name suggests, solve the model at fixed step sizes from the beginning to the end of the simulation. You can specify the step size or let the solver choose it. Generally, decreasing the step size increases the accuracy of the results and increases the time required to simulate the system.
     - Variable Step Solver :-
       Variable-step solvers vary the step size during the simulation. They reduce the step size to increase accuracy when the states of a model change rapidly and during zero-crossing events. They increase the step size to avoid taking unnecessary steps when the states of a model change slowly. Computing the step size adds to the computational overhead at each step. However, it can reduce the total number of steps, and hence the simulation time required to maintain a specified level of accuracy for models with piecewise continuous or rapidly changing states
   - Solver selection is depend on
     - Dynamics of the system
     - Stability of solution
     - Speed of computation

2) What is Step Size ?
   - Time steps are time intervals when the computation happens. The size of this time interval is called step size.

3) What is MATLAB ?
   - MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:
     - Math and computation
     - Algorithm development
     - Modelling, simulation, and prototyping

4) Why we use fixed step solver while generating code ?
   - Variable-step solver, the step size vary from step to step, depending on the model dynamics. In particular, a variable-step solver increases or reduces the step size to meet the error tolerances that we specify due to that the variable step sizes cannot be mapped to the real-time clock of a target system because the target system clock is fixed.
   - So due to we cannot mapped the real time clock of the target system so we not used variable step

solver while generating code and we used only fixed step solver for generating code.

5) What is Sample time , Simulation time, Start Time and Stop Time?
   - Sample Time ->
     - Sample time is a time taken by block to updates its internal state.
     - Sample time should be greater than 0 and less than simulation time
     - Sample times can be port based or block based.
     - For block-based sample times, all of the inputs and outputs of the block run at the same rate.
     - For port-based sample times, the input and output ports can run at different rates.
   - Simulation Time ->
     - It is the time defined by the user for the simulation of the model to verify the output as per the requirement.
   - Start Time ->
     - It is the time when the simulation of the model starts. We can define the start time as per our requirement.
   - Stop Time ->
     - it is the time when the simulation of the model stops.

   - Following are the types of Sample time:

   - Discrete Sample Time:-
     - Simulink compute block output only once at each of the fixed time interval eg. Unit delay block has discrete sample time
   - Continuous Sample Time:-
     - Unlike the discrete sample time, continuous sample times are divided into major time steps and minor time steps, where the minor steps represent subdivisions of the major steps.
     - The solver produces a result at each major time step. It uses results at the minor time steps to improve the accuracy of the result at the major time step.
     - To specify that a block, such as the Derivative block, is continuous, enter [0, 0] or 0 in the Sample time field of the block dialog.

   - Inherited Sample Time:-
     - If a block sample time is set to [–1, 0] or –1, the sample time is inherited and Simulink determines the best sample time for the block based on the block context within the model.
     - Simulink performs this task during the compilation stage the original inherited setting never appears in a compiled model. Therefore, you never see inherited ([–1, 0]) in the Sample Time Legend.

   - Constant Sample Time:-
     - In Simulink, a constant is a symbolic name or expression whose value you can change only outside the algorithm or through supervisory control. Blocks, like the constant block, whose outputs do not change during normal execution of the model, are always considered to be constant.
     - Simulink assigns constant sample time to these blocks. They run their block output method:
       - At the start of a simulation.
       - In response to runtime changes in the environment, such as tuning a parameter.
     - For constant sample time, the block sample time assignment is [inf,0] or [inf].
     - For a block to allow constant sample time, these conditions hold:
       - The block has no continuous or discrete states.

        - The block does not drive an output port of a conditionally executed subsystem (see Using Enabled Subsystems).
    - S-Function Blocks
        - The Simulink block library includes several blocks, such as the MATLAB S-Function block, the Level-2 MATLAB S-Function block, and the C S-Function block, whose ports can produce outputs at different sample rates. It is possible for some of the ports of these blocks to have a constant sample time.

          Tvo

- Variable Sample Time:-
    - Blocks that use a variable sample time have an implicit Sample Time parameter that the block specifies; the block tells Simulink when to run it. The compiled sample time is [–2,*Tvo*] where *Tvo* is a unique variable offset.
    - The Pulse Generator block is an example of a block that has a variable sample time. Since Simulink supports variable sample times for variable-step solvers only, the Pulse Generator block specifies a discrete sample time if you use a fixed-step solver.

- Triggered Sample Time:-
    - If a block is inside of a triggered-type (e.g., function-call, enabled and triggered, or iterator) subsystem, the block may be constant or have a triggered sample time.
    - You cannot specify the triggered sample time type explicitly. However, to achieve a triggered type during compilation, you must set the block sample time to inherited (–1). Simulink then determines the specific times at which the block computes its output during simulation.

6) What is algebraic loop error and how to resolve it?
    - An algebraic loop error occur when the output of a driven block is directly connected to the input of the same block.
    - This error can be resolve by connecting the unit delay block between the input and output signal. Unit delay is a memory block that delay and hold it’s input signal.

7) What is the difference between mux, merge and bus ?
    - Mux ->
        - mux block combine it's input into single vector output.
        - All input must be same data type and numeric type.
        - The mux block accept real or complex data type that Simulink support including fixed and enumerated data types.(mux block accept any type of data type that Simulink Support).
    - Merge ->
        - Merge block combine all its inputs from conditionally executed subsystem into single output line whose value at any time is equal to the most recently computed output value of conditionally executed subsystem block.
        - Merge block assume all driving signal sharing same signal memory.
        - All the input of merge block have same sample time.
        - Merge block only accept input signal from conditionally executed subsystem.
    - Bus ->
        - The bus creator block combine a set of input signal into bus.
        - We can connect any signal type to bus creator block including other buses signal.
        - We can access signal in bus by using bus selector block.
        - We can use array of buses as an input to the bus creator block.
        - To ungroup bus signals, connect the output of bus creator block to bus selector block.

8) What is Real Time System ?
   - A real-time operating system (RTOS) is an operating system (OS) intended to serve real-time applications that process data as it comes in, typically without buffer delays.
   - Processing time requirements (including any OS delay) are measured in tenths of seconds or shorter increments of time.
   - A real-time system is a time-bound system which has well-defined, fixed time constraints.
   - Processing must be done within the defined constraints or the system will fail
   - Simulink real time lets you create real-time application from simulink models and run them on target computer hardware connected to your physical system.

9) What is model based design and its use ?
   - Model-based design (MBD) is a mathematical and visual method of solving problems associated with designing complex control System, signal processing and communication systems.
   - It is used in many motion control, industrial equipment, aerospace, and automotive applications. Model-based design is a methodology applied in designing embedded software.
   - MBD Allows :-
     - Common design environment
     - Linking design directly to requirement
     - Generating embedded software code

10) Shortcuts
   - Create subsystem --> ctrl + G
   - Create mask --> ctrl + M
   - Look under mask --> ctrl + u

11) Available platform for MBD in Automotive
   - MATLAB/Simulink

     Other Tools
     - Octave
     - LabVIEW
     - Multisim
     - Proteus
     - Maple soft
     - Scilab/scicos

12) Datatypes in Simulink

Double	64 bit	
Single	32 bit	
Int8	8 bit	-127 to +126
Uint8	8 bit	0 to 255
Int16	16 bit	-32768 to +32767
Uint16	16 bit	0 to 65535
Int32	32 bit	
Uint32	32 bit	
Boolean	0/1 bit	

Int64	64 bit	
Uint64	64 bit	

13) Fixed point tool
   - Create a model ---> create subsystem of model
   - Log signal ---> run model
   - Right click ---> fixed point tool(Analysis)

15) For different data type which block we can use ?
   - if the input to the block of different data type then we can use bas creator block.

16) Why to use Simulink over other development tools ?
   - Same environment of development and testing
     - so you don't have to switch between tools to test your software. So you develop the model instead of generation the code or testing it other environment or actually building the code and flashing it on hardware you first write mil test cases in signal builder or write test cases in excel sheet and import to test vectors in signal builder due to this you resolve issue related to software in initial development of the software so when you go to the hardware testing most of the issue are related to hardware or related to configuration but the software or logic related issue get resolve in initial development stage
   - Auto code generation
     - we don't have to write code manually we just need to focus on logic development, due to auto code generation it generate standard code so the coding which varies person to person that is avoided.
   - graphical design
     - because visualization of logic is better than reading of thousand line of code
       we can easily understand what logic is doing .

17) How to convert 2 or more signal in to vector ?
   - Using mux block we can convert 2 or more signals into vector.

19) How you will decide in between Stateflow and Simulink, when to use on what basis?
   - The selection of the Stateflow and the Simulink depends on the requirement from the client.
   - If client have mention any particular tool to use for the development of the model then we will go with that option.

     OR
   - If the requirement of the client contain complex model then we will go with the Stateflow and if not then we can use the Simulink.
   - Simulink is used for functional algorithm modelling which depicts the graphical math operations like product, sum and integrals.
   - When the model contain logical and state based system then we can go with the Stateflow.

20) How many types of Workspace we have in MATLAB and explain them?
   - Base Workspace:
     - It is the workspace that store the variable created in the MATLAB command window.
     - To access the variable from this workspace we need to save the variable as the .m file for further use.
   - Model Workspace:
     - This is the workspace that store the variable used for the model. From here we can use the

created variable directly into our model environment.
  - Function Workspace:
    - This is the workspace used by the function. When we create the function automatically this workspace gets created and the variable that are used in the function get stored in it. Once the function get deleted this workspace also gets vanished.

21) Which block can be used to change the rate of the signal?
  - If you want to change rate within a system, use a rate transition block which is designed specifically to model rate transition.

22) In what form you get the requirement from the client?
  - We get the requirement in the form of word, excel, pdf and html file.

23) What is virtual and non virtual bus?
  - A virtual bus is just a visual representation in the Simulink editor. It makes your model look nicer and easier to manage when you have many signals.
  - We used virtual bus to group the signal.
  - It does not affect how the signal is stored in memory and It has no impact on the code of the model.
  - Non Virtual bus is analogous to a struct in C code. the data is stored as a structure in a contiguous piece of memory. Because of that, nonvirtual have more constraints.

24) When do you need non-virtual bus ?
  - The only places we need nonvirtual buses are at the boundaries in our model when we want to define the interface as a structure.
  - For example of these boundaries are Stateflow charts or Matlab Function.

25) What are the Merge Guideline?
  - Always use conditionally-executed subsystems to drive Merge blocks.
  - when you write control logic to ensure that at least one of the driving conditionally-executed subsystems executes at any one time step.
  - Always connect a Merge block to at least two input signals.
  - Ensure that all input signals have the same sample time.
  - Always set the Initial output parameter of the Merge block, unless the output port of the Merge block connects to another Merge block.
  - Do not branch a signal that inputs to a Merge block.
  - For all conditionally-executed subsystem Out port blocks that drive Merge blocks, set the Output when disabled parameter to held.
  - If the output of a Model block is coming from a MATLAB Function block or a Stateflow chart, do not connect that output port to the input port of the Merge block.

26) Swap the two value from script.
  - First create a script file whose extension will be .m file

```
a = a + b;
b = a - b;
a = a - b;
```

  - Save this file and run it, now in command window

```
a = 20; b = 40;
```

  - Run the above script file. Once it is run we can see the value of a and b get swap in the base workspace.

27) How to get the input from the workspace to model environment and model to workspace?
    - From Workspace Block and To Workspace Block.

28) How to change the rate of signal?
    - We can use the rate transition block to change the rate of the signal.

29) Which block generate switch case code in c ?
    - multiport switch.

30) System Mask.
    - A model consists of multiple blocks, with each block containing its own parameter and block dialog box.
    - Simulink® enables us to mask a model. By masking a model we encapsulate the model to have its own mask parameter dialog box.
    - we can customize the mask parameter dialog box. When we mask a model, the model arguments become the mask parameters.
    - Referencing a masked model helps us having a better user interface for a model with ease of controlling the model parameters through the mask.

31) How to convert virtual signal to non-virtual signal ?
    - using signal conversion block we can convert virtual signal to non-virtual signal.

32) Fixed point number.
    - The dynamic range of fixed-point values is less than floating-point values with equivalent word sizes. To avoid overflow and minimize quantization errors, fixed-point numbers must be scaled.
    - We have two method to scale fixed point number

    - Slop Bias Scaling
        - You can represent a fixed-point number by a general slope and bias encoding scheme. The real world value of a slope bias scaled number can be represented by:

          real-world value = (slope × integer) + bias
        - The slope and bias together represent the scaling of the fixed-point number.
        - In a number with zero bias, only the slope affects the scaling. A fixed-point number that is only scaled by binary point position is equivalent to a number in slope bias representation that has a bias equal to zero and a slope adjustment factor equal to one. This is referred to as binary point-only scaling or power-of-two scaling.

    - Binary-Point-Only Scaling =
        - Binary-point-only or power-of-two scaling involves moving the binary point within the fixed-point word.
        - The advantage of this scaling mode is to minimize the number of processor arithmetic operations.

33) What is S - Function and why we used it ?
    - S-function lets us bring c-code in Matlab environment so when we have mathematical equation for our code and instead of developing model of same c code using different block we want one

general purpose block then we can use s-function.

- Procedure to generate s-function Using legacy tool =

  Def = legacy_code('initialize')   // it will initialize structure
  Def.SFunctionName = 'New_s_func' // write here s - function name
  Def.OutputFcnSpec = 'double y1 = add_2(double u1,double u2)' // Used y1 for output and u1 u2 for input other than that it throw error
  Def.HeaderFiles = {'add_lib.h'} //Input library header file
  Def.SourceFiles = {'main.c'} //Add main c file
  Legacy_code('sfcn_cmex_generate',def) //it generate new c code for s-function
  Legacy_code('compile',def) //compile generated c file
  After this procedure add s-function block in new Simulink editor window and select s-function name and connect the input and output to that block.

34) S-Function Builder.

- first add the s-function builder block to simulink editor and open it then go to the data property add inputs and outputs, add s-function name select target language c and then build the s-function after that open matlab command window write mex matlab command followed by file name and file wrapper function name and then enter after that s-function will get created.

35) Benefits of Using fixed point Hardware ?

- Within digital hardware, numbers are represented as either fixed-point or floating-point data types. For both of these data types, word sizes are fixed at a set number of bits.
- However, the dynamic range of fixed-point values is much less than floating-point values with equivalent word sizes. Therefore, in order to avoid overflow or unreasonable quantization errors, fixed-point values must be scaled.
- Since floating-point processors can greatly simplify the real-time implementation of a control law or digital filter, and floating-point numbers can effectively approximate real-world numbers, then why use a microcontroller or processor with fixed-point hardware support?
- Size and Power Consumption
  - The logic circuits of fixed-point hardware are much less complicated than those of floating-point hardware. This means that the fixed-point chip size is smaller with less power consumption when compared with floating-point hardware. For example, consider a portable telephone where one of the product design goals is to make it as portable (small and light) as possible. If one of today's high-end floating-point, general-purpose processors is used, a large heat sink and battery would also be needed, resulting in a costly, large, and heavy portable phone.
- Memory Usage and Speed
  - In general fixed-point calculations require less memory and less processor time to perform.
- Cost
  - Fixed-point hardware is more cost effective where price/cost is an important consideration. When digital hardware is used in a product, especially mass-produced products, fixed-point hardware costs much less than floating-point hardware and can result in significant savings.

36) Simulink Design Verifier

- We used simulink design verifier to Identify design errors, check requirements compliance, and generate tests
- Simulink® Design Verifier™ identify hidden design errors in models. It detects blocks in the model that result in integer overflow, dead logic, array access violations, and division by zero error. It can

formally verify that the design meets functional requirements. For each design error or requirements violation, it generates a simulation test case for debugging.

- Simulink Design Verifier generates test cases for model coverage and custom objectives to extend existing requirements-based test cases. These test cases drive your model to satisfy condition, decision, modified condition/decision (MCDC), and custom coverage objectives. In addition to coverage objectives, you can specify custom test objectives to automatically generate requirements-based test cases.
- Steps :-
  - Make a subsystem atomic
  - Right click on subsystem and click on design verifier
  - Hear you can check different scenario compatibility of the subsystem
  - You can able to check the design errors in your subsystem
  - You will be able to generate test cases for your subsystem
  - You can prove your subsystem properties in your subsystem by using design verifier.
  - In subsystem inport and outport should be taken
  - Log signals
  - Generate test cases
  - Simulink test --> export test cases ---> create harness
  - Simulated test cases to simulink test

37) What is the datatype of bus creator block ?
- Datatype of bus creator block is struct

38) How to do root inport mapping ?
- Define matrix variable
  Eg. U = [1 2 3 4 5]
  Ds = mat2dataset(U)
  - Go to port block parameter
  - Then go to signal attribute
  - Select datatype to double, port dimension to 1 and signal type to real
  - Then select connect input option root inport mapping tool get open
  - Then you can add input element via spreadsheet, mat file or workspace
  - Then select port order and select map to model option

39) Root inport mapping by using simulink timeseries.
- Create a model having input port
- Go to command window and write command related to simulink time series
  A = Simulink.Timeseries
  A.Time = [1 2 3 4]
  A.Data = [3 1 3 2]
  - Go to port block parameter
  - Then go to signal attribute
  - Select datatype to double, port dimension to 1 and signal type to real
  - Then select connect input option root inport mapping tool get open
  - Then you can add input element via spreadsheet, mat file or workspace
  - Then select port order and select map to model option

40) Difference between scripting and function.

Scripting	function

Scripting m file can only operate On the variable that are assigned	Function m file having a input and Output
Script are useful for task that don't Change	Function are more flexible and more suitable For general purpose
Script variable are global	Function variables are local to the function. Local scope of function variable gives you grater security and flexibility
We can directly assign value to scripting variable in workspace	The only way to set information into and out of a function is through the variable in the parameter list

41) What is MATLAB coder, Simulink coder and Embedded coder ?

- MATLAB Coder
  - MATLAB® Coder™ generates C and C++ code from MATLAB code for a variety of hardware platforms, from desktop systems to embedded hardware. It supports most of the MATLAB language and a wide range of toolboxes. we can integrate the generated code into our projects as source code. The generated code is readable and portable. we can combine it with key parts of our existing C and C++ code and libraries. we can also package the generated code as a MEX-function for use in MATLAB.
- Simulink Coder
  - Simulink® Coder™ (formerly Real-Time Workshop®) generates and executes C and C++ code from Simulink models, Stateflow® charts, and MATLAB® functions. The generated source code can be used for real-time and non-real-time applications, including simulation acceleration, rapid prototyping, and hardware-in-the-loop testing. we can tune and monitor the generated code using Simulink or run and interact with the code outside MATLAB and Simulink.
- Embedded Coder
  - Embedded Coder® generates readable, compact, and fast C and C++ code for embedded processors used in mass production. It extends MATLAB® Coder™ and Simulink® Coder with advanced optimizations for precise control of the generated functions, files, and data. These optimizations improve code efficiency.
  - Embedded Coder offers built-in support for AUTOSAR and MISRA C® software standards.

42) What is fixed point and floating point number ?

- Fixed point
  - In fixed point number there are fixed number of digit after decimal point

    Example –Assume number is using 32-bit format which reserve 1 bit for the sign, 15 bits for the integer part and 16 bits for the fractional part.

    Then, -43.625 is represented as following:

1	000000000101011	1010000000000000
Sign bit	Integer part	Fractional part

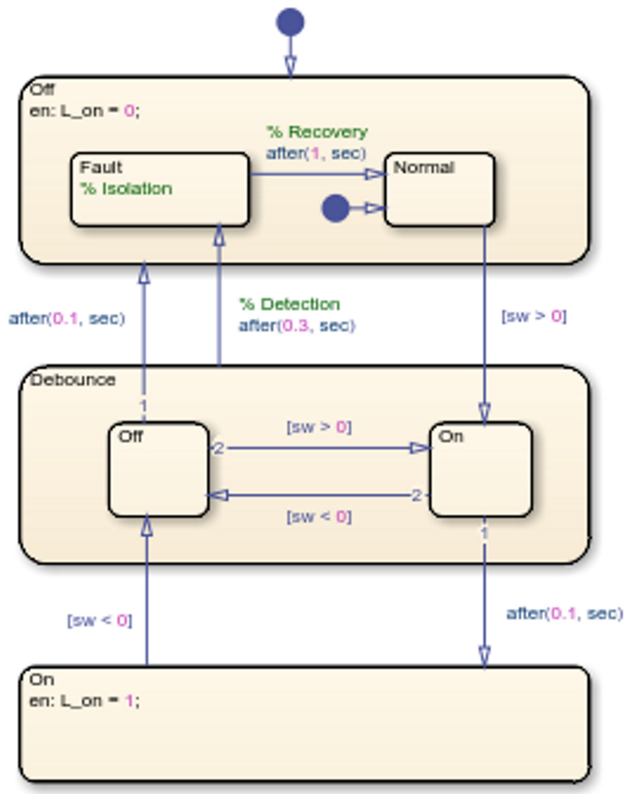
  - The advantage of using a fixed-point representation is performance As the value stored in memory is an integer
  - and disadvantage is relatively limited range of values that they can represent. So, it is usually inadequate for numerical analysis as it does not allow enough numbers and accuracy. A number whose representation exceeds 32 bits would have to be stored inexactly.
  - However, there is a downside! Fixed Point Representations have a relatively limited range of values

that they can represent.

- Floating point
  - In floating point number allow for a varying number of digit after the decimal point
  - Floating Point Notation is an alternative to the Fixed Point notation and is the representation that most modern computers use when storing fractional numbers in memory. Floating Point Notation is a way to represent very large or very small numbers precisely using scientific notation in binary. In doing so, Floating Point Representation provides a varying degrees of precision depending on the scale of the numbers that you are using.
  - The advantages of floating point is ranges of number is increased also accuracy gets increases.
  - And disadvantage is performance of hardware get decreases.
  - More memory required.

43) Multiply 21 with 4 without using multiply block.
  - We can use the gain block to multiply 21 with 4.

44) What is debouncing logic ?
  - When a switch opens and closes, the switch contacts can bounce off each other before the switch completely transitions to an on or off state. The bouncing action can produce transient signals that do not represent a true change of state. Therefore, when modelling switch logic, it is important to filter out transient signals by using debouncing algorithms.
  - If you model a controller in a Stateflow® chart, you do not want your switch logic to overwork the controller by turning it on and off in response to every transient signal it receives. To avoid this, design a Stateflow controller that uses temporal logic to debounce your input signals and determine whether a switch is actually on or off.



45) Storage Classes in Matlab ?

- Storage classes help us to integrate generated code with external code.
- we can make a generated variable visible to external code. we can also make variables declared in the external code visible to the generated code.
- For code generation from MATLAB® code, we can use storage classes with global variables only. The storage class determines:
- The file placement of a global variable declaration and definition.
- Whether the global variable is imported from external code or exported for use by external code.

Storage Class	Description
'ExportedGlobal'	• Defines the variable in the Variable Definitions section of the C file *entry_point_name*.c. • Declares the variable as an extern in the Variable Declarations section of the header file *entry_point_name*.h • Initializes the variable in the function *entry_point_name*_initialize.h.
'ExportedDefine'	Declares the variable with a #define directive in the Exported data define section of the header file *entry_point_name*.h.
'ImportedExtern'	Declares the variable as an extern in the Variable Declarations section of the header file *entry_point_name*_data.h. The external code must supply the variable definition.
'ImportedExtern Pointer'	Declares the variable as an extern pointer in the Variable Declarations section of the header file *entry_point_name*_data.h. The external code must define a valid pointer variable.

46) What is MEX file in MATLAB ?
- A MEX file is a type of computer file that provides an interface between MATLAB and functions written in C, C++. It stands for "MATLAB executable".
- A MEX file is a function, created in MATLAB, that calls a C/C++ program subroutine. A MEX function behaves just like a MATLAB script or function.

47) Simulink Vs. Stateflow ?
- Simulink is used to respond to continuous changes in dynamic changes. Stateflow is used to respond to instantaneous changes in dynamic changes. Real-world systems have to respond to both continuous and instantaneous changes. Use both Simulink and Stateflow so that you can use the right tool for the right job

48) Quantization error.
- Quantization is the process of mapping continuous infinite values to a smaller set of discrete finite values. In the context of simulation and embedded computing, it is about approximating real-world values with a digital representation that introduces limits on the precision and range of a value. Quantization introduces various sources of error in your algorithm, such as rounding errors, underflow or overflow, computational noise, and limit cycles. This results in numerical differences between the ideal system behaviour and the computed numerical behaviour.
- To manage the effects of quantization, you need to choose the right data types to represent the real-world signals. You need to consider the precision, range, and scaling of the data type used to encode the signal, and also account for the non-linear cumulative effects of quantization on the numerical behaviour of your algorithm. This cumulative effect is further exacerbated when you have constructs such as feedback loops.

49) What is difference between Simulink and Stateflow ?

- Simulink is a block diagram environment for multi domain simulation and Model-Based Design. It supports system-level design, simulation, automatic code generation, and continuous test and verification of embedded systems.
- Simulink provides a graphical editor, customizable block libraries, and solvers for modelling and simulating dynamic systems.
- Stateflow is an environment for modelling and simulating combinatorial and sequential decision logic based on state machines and flow charts. Stateflow lets you combine graphical and tabular representations, including state transition diagrams, flow charts, state transition tables, and truth tables, to model how your system reacts to events, time-based conditions, and external input signals.
- Simulink is used for functional algorithm modelling whereas stateflow is used for logical for logical and state based system.
- Simulink is largely a control oriented solution whereas stateflow deals extremely well with logic and of course state machine.
- Simulink graphically depicts math like products, sums, integrals etc whereas stateflow deals with the model which involve the if construct logic.
- Simulink is used to response to continuous changes in dynamic changes whereas stateflow is used to respond to instantaneous changes in dynamic changes.

50) Stateflow Vs. Flowchart

Stateflow	Flowchart
• Stateflow is use for logical and state based system	• Flowchart is a graphical construct that model logic pattern by using connective junction and transition
• Default transition execute in stateflow only single time	• Default transition execute in flowchart in every single time
• Stateflow contain a states	• Flowchart contain transition path and junction and do not contain states
• Stateflow consider the previous result output	• Flowchart not consider the previous result output