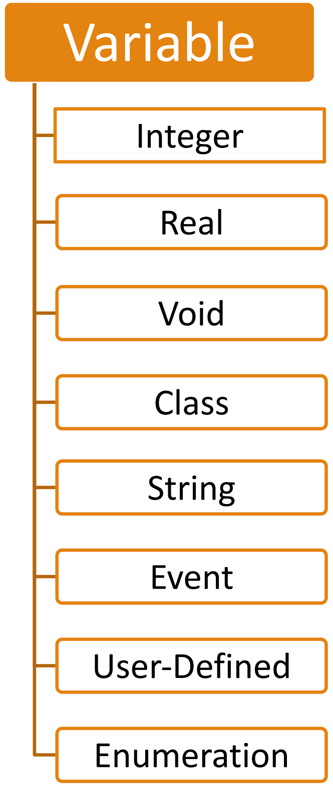
SystemVerilog for Verification

BASIC DATA TYPES – PART II

Agenda

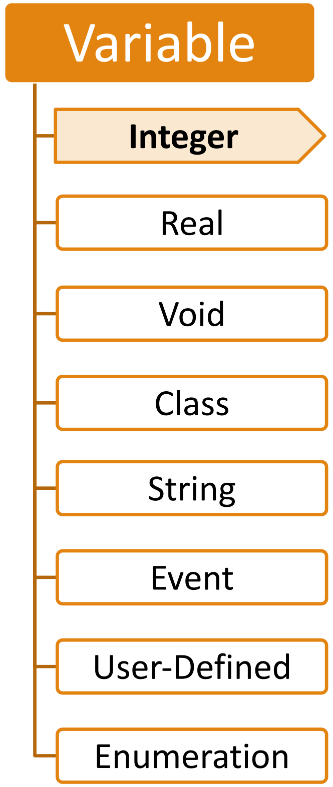


Variable

- Integer
- Real
- Void
- Class
- String
- Event
- User-Defined
- Enumeration

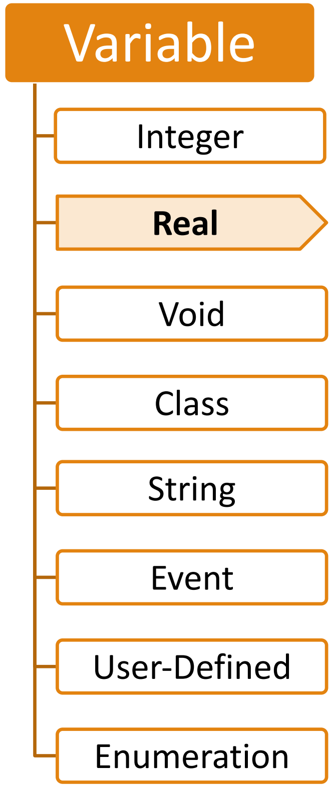
Variable – a data storage element

logic [3:0] abcd;



Integer	State value	Size	Sign	Default Value
shortint	2	16 bits	signed	‘0
int	2	32 bits	signed	‘0
longint	2	64 bits	signed	‘0
byte	2	8 bits	signed	‘0
bit	2	user-defined vector	unsigned	‘0
logic	4	user-defined vector	unsigned	‘X
reg	4	user-defined vector	unsigned	‘X
integer	4	32 bits	signed	‘X
time	4	64 bits	unsigned	‘X

Variable – a data storage element



Variable

Integer

Real

Void

Class

String

Event

User-Defined

Enumeration

Real type (floating point)	C-type	Size	Sign	Default Value
real	double	64 bits	signed	0.0
shortreal	float	32 bits	signed	0.0
$realtime				

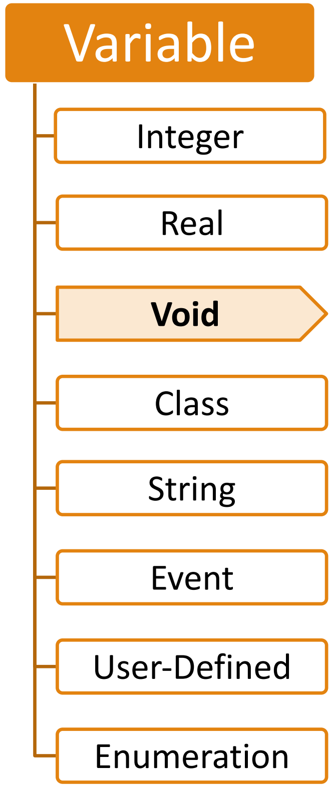
$realtime vs $time – Depends on timescale

Exercise Time

1. Perform addition on bit & integer type operands, logic & bit type operands. Assign four state initial value (containing x,z,1,0) to all four types of operand... Print their initial values as well as result after addition. See the effect of data types.

2. Assign {32{4'b1111}} to bit, byte, shortint, int, longint and integer data types and print them. Repeat the same thing with "unsigned" declaration for all above data types.. Use four state value {32{4'b01xz}} and repeat the above steps.

Variable – a data storage element



Variable

Integer

Real

Void

Class

String

Event

User-Defined

Enumeration

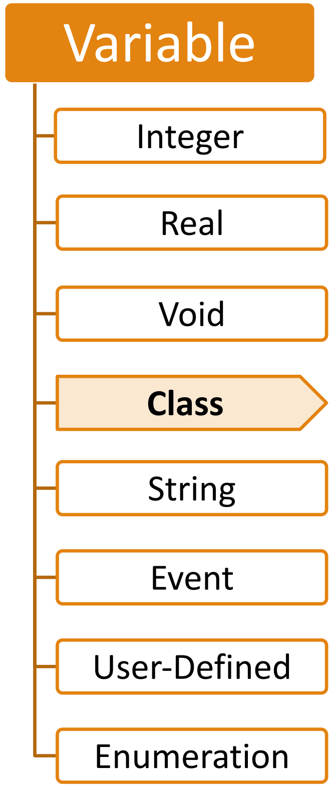
Void – nonexistence of data

functions indicating no return type

void = function_call();

type of tagged union

Variable – a data storage element


Variable

- Integer
- Real
- Void
- **Class**
- String
- Event
- User-Defined
- Enumeration

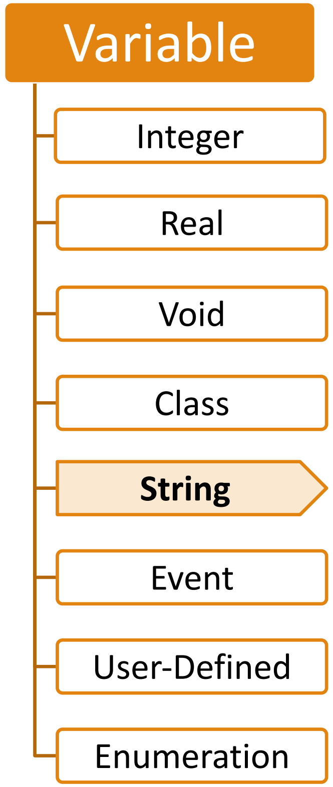
OOPS

Can contain *properties* and *methods*

```
class data;
   bit [3:0] abc;
   logic [4:0] cdf;
   integer pqr;
   task clean();
      abc = 4’b0; cdf = 5’b0; pqr = 0;
   endtask
endclass
```

default value = null

Variable – a data storage element



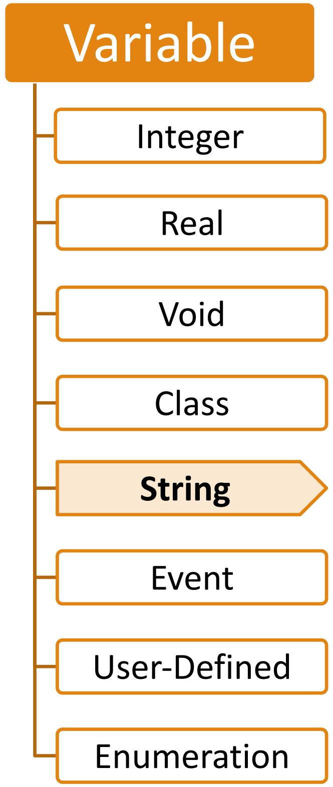
➢ variable size, dynamically allocated array of bytes

string s0 = “Hello World”; byte s1 [0:10] = “Hello World”;

s0 = {s0, “new”}; → s0 : “Hello World new”

Operator	Semantics
str1 == str2 , str1 != str2	equality
str1 > str2 , >= , < , <=	comparison
{str1, str2}	concatenation
{multiplier{str1}}	replication
str[index]	indexing
str.method()	methods onto strings

Variable – a data storage element



Variable

Integer

Real

Void

Class

String

Event

User-Defined

Enumeration

Method	Description
Str.len()	Returns length of string
Str.putc(int i, string s)	Replaces ‘i’th char in string with first char in s
Str.getc(int i)	Returns the ASCII code of the ‘i’th char in str
Str.toupper()	Returns a string with chars in string converted to upper case. Source string remains unchanged
Str.tolower()	Returns a string with chars in string converted to lower case. Source string remains unchanged

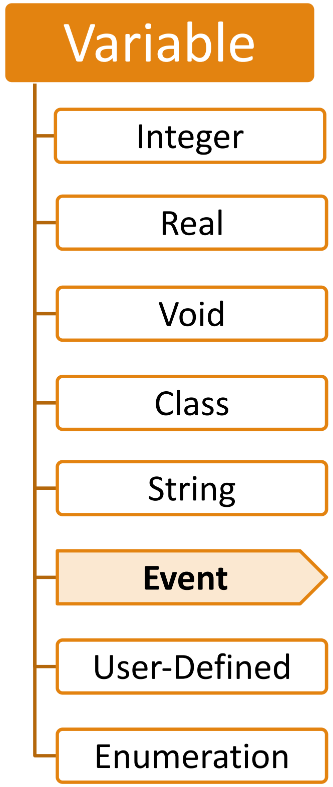
str1.compare(str2)
str1.icompare(str2)
	Compares str1 with str2 (non case sensitive)
	Compares str1 with str2 (case sensitive)

Str.substr(i,j)	Returns new string that is a substring formed by characters in position I through j of str.

String Exercise

- Write 2 string variables, str1=Hello and str2=World.
- Print the str1 and size of str1;
- Declare new string variable New_Str, concatenate str1 and str2 and assign it to New_Str
- Print the New_Str and size of New_Str
- Declare new string variable NEW_STR, call the New_Str.toupper() and assign the returned string to the NEW_STR variable and then print the NEW_STR
- Similarly call the tolower() method, assign the returned string to new_str variable and print it.
- Try other string operations

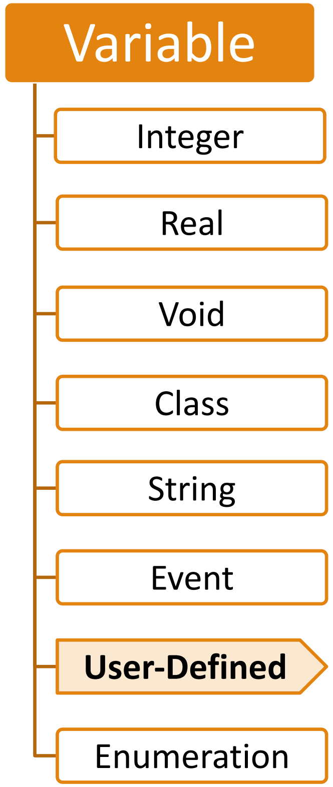
Variable – a data storage element



- ➢ synchronization of, two or more concurrently active processes.

```
event a;    // declaration
→ a;        // event triggerred
@(a);       // waiting for occurrence of the event trigger
```

Variable – a data storage element



Variable

Integer

Real

Void

Class

String

Event

User-Defined

Enumeration

Typedef

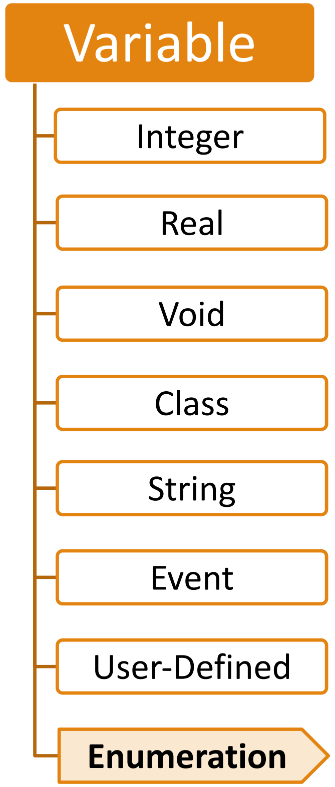
typedef data_type type_identifier **;**

Ex.

typedef int animal;

animal lion, tiger;

Variable – a data storage element



- a set of integral named constants

```
enum {red, yellow, green} ligh
```

← red = 0, yellow = 1, green = 2

```
enum bit [1:0] {IDLE, XX='x, S1=2'b01, S2=2'b10}
state, next;
```

← Syntax error

```
enum integer {IDLE, XX='x, S1='b01, S2='b10}
state, next;
```

← IDLE = 0, others having values

```
enum {bronze=3, silver, gold} n
```

← silver = 4, gold = 5

```
enum {a=3, b=7, c} alphak
```

← c = 8

Enum Exercise

Design a sequence detector ‘1011’ using state machine. Declare present_state & next_state as enum.

```
Hint : typedef enum logic [1:0] {s0,s1,s2,s3} state;
       state present_state, next_state;
```

Next Session – Parameter

– Scope & Lifetime

– Casting