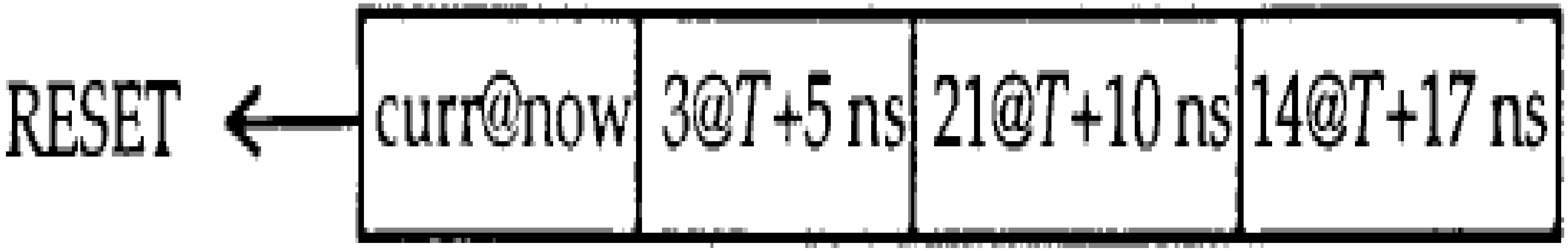
Signal Drivers

- **What if there is more than one assignment to the same signal within a process?**
- Every signal assignment in a process creates a *driver for that signal.*
- *The driver of a signal holds its current value and all its future values as a sequence of one or more transactions, where each transaction identifies the value to appear on the signal along with the time at which the value is to appear.*

Signal Drivers Cont..

- Consider the following signal assignment statement with three waveform elements and the driver that is created for that signal.
- **process**
- **begin**
    - RESET **<= *3 after 5 ns, 21 after 10 ns, 14 after 17 ns;***
- **end process;**

Signal Drivers Cont..

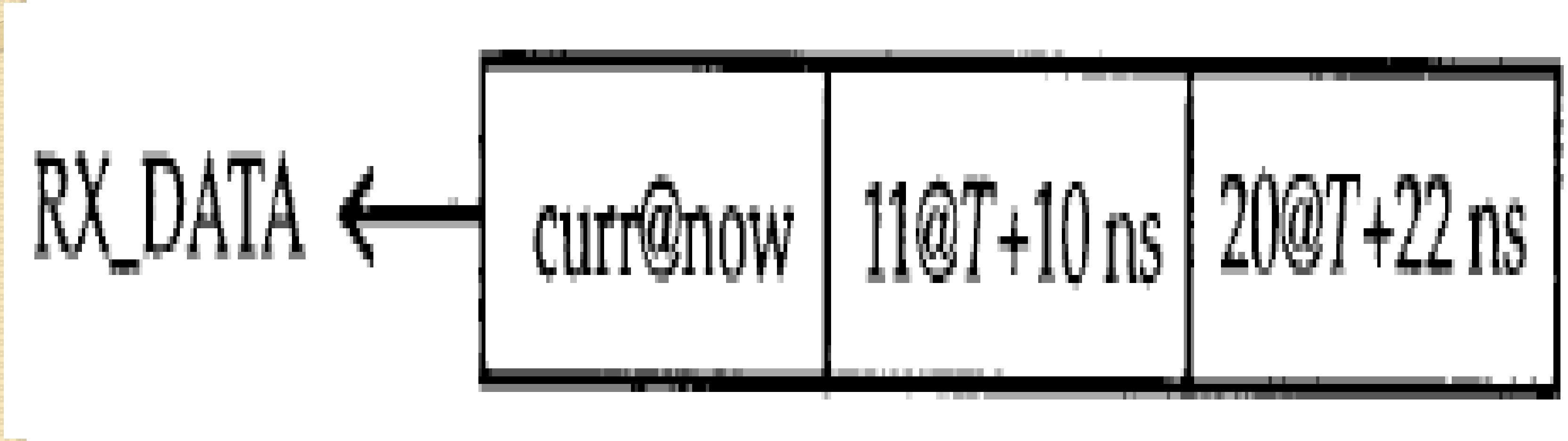


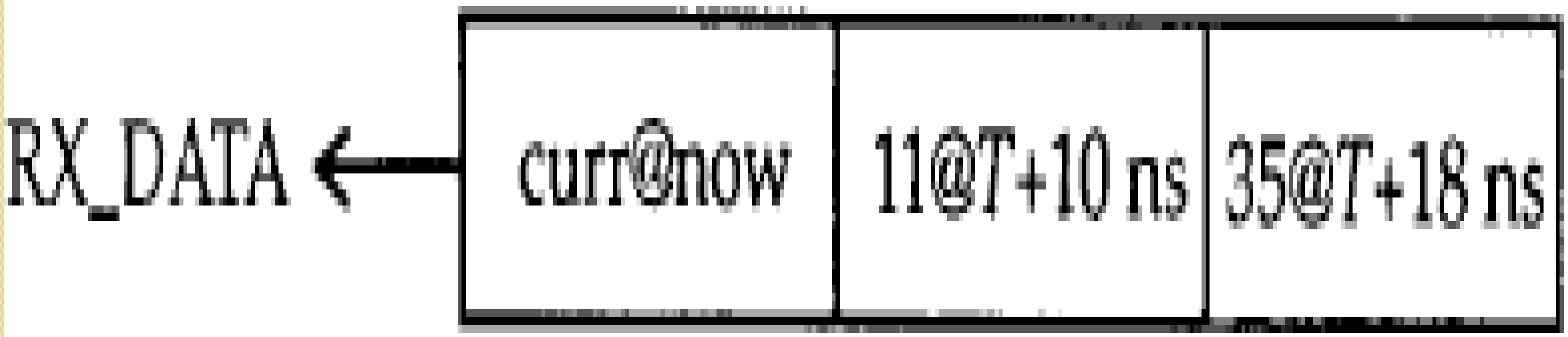
Effect of Transport Delay on Signal Drivers

- Here is an example of a process having three signal assignments to the same signal RX_DATA.

```
signal RX_DATA: NATURAL;
……
process
begin
   RX_DATA <= transport 11 after 10 ns:
   RX_DATA <= transport 20 after 22 ns;
   RX_DATA <= transport 35 after 18 ns;
end process;
```

Effect of Transport Delay on Signal Drivers cont..





Effect of Transport Delay on Signal Drivers cont..

- Here is another example.

process

--DATA_BUS is a signal of type BIT_VECTOR.

begin

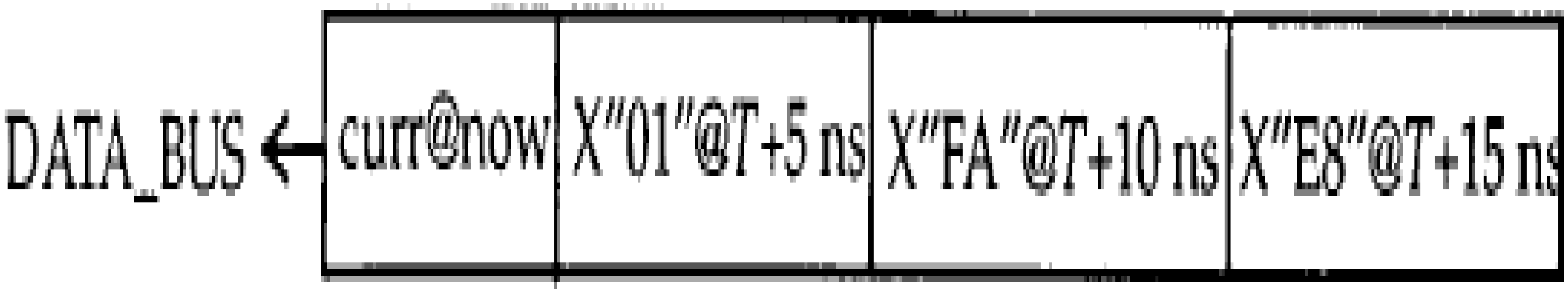
DATA_BUS **<= transport X"01" after 5 ns, X"FA" after 10 ns,**

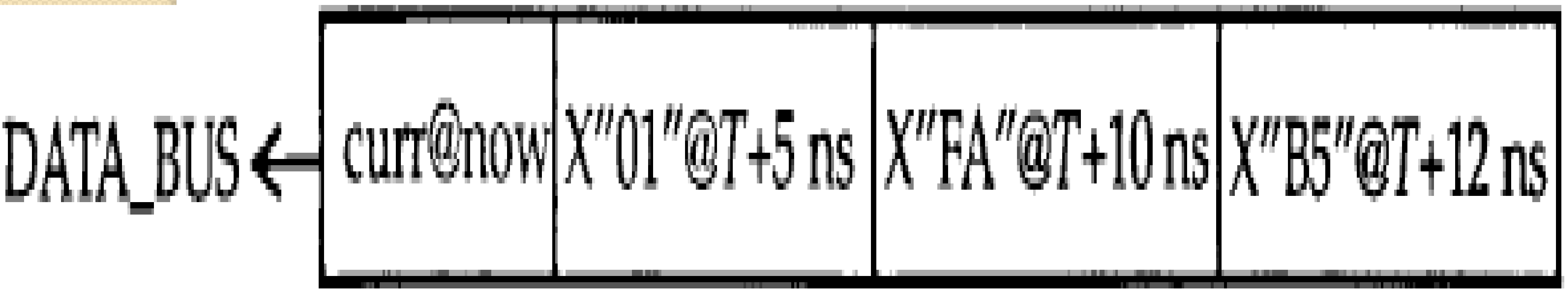
X"E8" **after 15 ns;**

DATA_BUS <= ***transport X"B5" after 12 ns;***

end process;

Effect of Transport Delay on Signal Drivers cont..





Effect of Transport Delay on Signal Drivers cont..

- The following summarizes the rules for adding a new transaction to a driver in the transport delay mode.
- 1. If the delay time of the new transaction is greater than those of all the transactions already present on the driver, then the new transaction is added at the end of the driver.
- 2. If the delay time of the new transaction is earlier than or equal to one or more transactions on the driver, then these transactions are delete from the driver and the new transaction is added at the end of the driver.

Effect of Inertial Delay on Signal Drivers

- When inertial delays are used, both the signal value being assigned and the delay value affect the deletion and addition of transactions.

- If the delay of the new transaction is earlier than an existing transaction, the latter is deleted and the new one is added at the end of the driver, regardless of the signal values of the two transactions.

Effect of Inertial Delay on Signal Drivers cont..

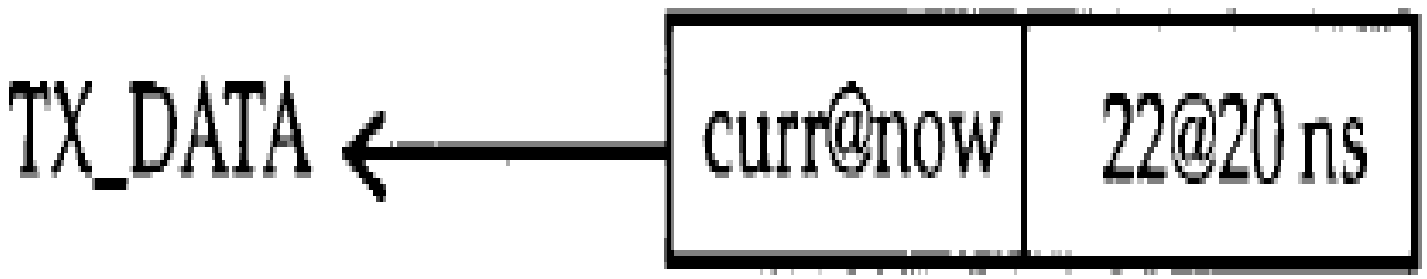
- On the other hand, if the delay of the new transaction is greater than an already existing one, the signal values of the two transactions are compared.
- If they are the same, the new transaction is simply added at the end of the driver, if not, the existing one is deleted before adding the new transaction. Deletion occurs for every existing transaction with a signal value that is different from the new transaction.

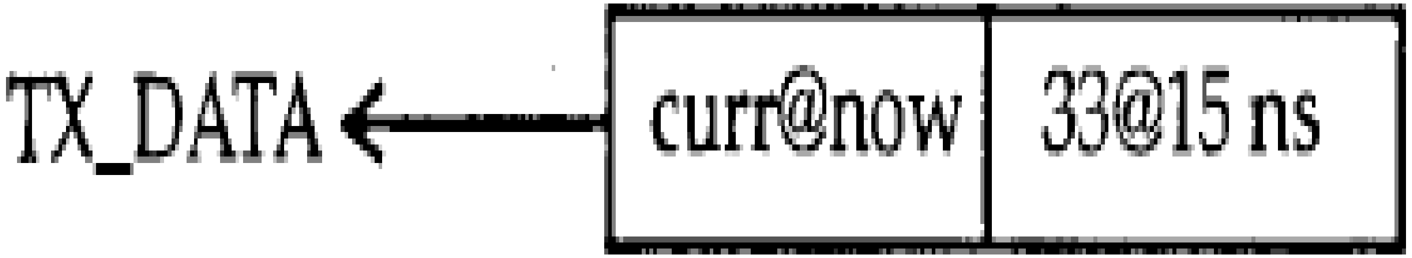
Effect of Inertial Delay on Signal Drivers cont..

- Consider the following process statement.

```
process
begin
   TX_DATA <= 11 after 10 ns;
   TX_DATA <= reject 15ns inertial 22 after
     20 ns;
   TX_DATA <= 33 after 15 ns;
   wait; -- Suspends indefinitely.
end process;
```

Effect of Inertial Delay on Signal Drivers cont..





Effect of Inertial Delay on Signal Drivers cont..

- Let us consider another example.

```
process
begin
  ADDR_BUS <= 1 after 5 ns, 21 after 9 ns, 6
     after 10ns,12 after 19 ns;
  ADDR_BUS <= reject 4ns inertial 6 after 12 ns,
     20 after 19 ns;
wait;
end process;
```

Effect of Inertial Delay on Signal Drivers cont..

- The summary of rules for adding a new transaction when inertial delay is used is
- 1. All transactions on a driver that are scheduled to occur at or after the delay of the new transaction are deleted (as in the transport case).
- 2 Add all the new transactions to the driver.

Effect of Inertial Delay on Signal Drivers cont..

3. For all the old transactions on the driver that occur at times between the time of the first new transaction(say F) and F minus the pulse rejection limit, delete the old transactions whose value is different from the value of the first new transaction.

Operator

Precedence	Operator Class	Operators					
Low	Logical	AND	OR	NAND	NOR	XOR	XNOR
↓	Relational	=	/=	<	<=	>	>=
	Shift	sll	srl	sla	sra	rol	ror
	Add	+	-	&			
	Sign	+	-				
	Multiply	*	/	mod	rem		
High	Miscellaneous	**	abs	not			

Logical Operator

- AND
- OR
- NAND
- NOR
- XOR
- NOT

Operate on bit BUT can be overloaded for operation Bit_vector

- defined for the predefined types BIT and BOOLEAN.

Relational Operator

- **Compare the two values of the same base and return a Boolean value**
- **= (Equal)**
- **=/ (Not equal)**
- **> (Greater than)**
- **>= (Greater than equal to)**
- **< (Less than)**
- **<= (Less than equal to)**
- all relational operations is always BOOLEAN.

Shift Operators

- Sll (Shift Left logical)
- Srl (Shift Right Logical)
- Sla (Shift Left Arithmetic)
- Sra (Shift Right Arithmetic)
- Rol (Rotate Left)
- Ror (Rotate right)

  - A= 00001111

Adding Operator

- ◦ **+**
- ◦ **-**
- ◦ **&**

- **Operators '+' and '-'are predefined for integer operands can be overloaded for operation on data of the type bit_vector.**
- **Concatenation operator '&' is predefined for one-dimensional array.**
  - ***SIGNAL A : BIT_VECTOR (0 TO 3);***
  - ***SIGNAL B : BIT;***
  - ***SIGNAL C : BIT_VECTOR(0 TO 4);***
  - ***C<= B & A;***
- ***IF A="0000" AND B = '1' THEN C ="10000"***

Multiplying Operator

- *** (Multiply)**
- **/ (Divide)**
- **Rem**
- **Mod**
- predefined for both operands being of the same integer or floating point type.
- The result of a rem operation has the sign of its first operand and is defined as
- A **rem B = A - (A / B) * B**
- The result of a mod operator has the sign of the second operand and is defined as
- A **mod B = A – B * N**

Miscellaneous Operator

- **** (Exponent)**
- **Abs (absolute)**
- **Not**
- The **abs (absolute) operator is defined for any numeric type.**
- The **** (exponentiation) operator is defined for the left operand to be of integer or floating point type and the right operand (i.e., the exponent) to be of integer type only.**
- The not logical operator has the same precedence as the above two operators.