ScalaHDL

by Yao Li

Outline

- Gates and FSM.
- Slicing and concatenation.
- Unsigned/Signed types.
- Future Plan.

Gates and FSM: Gates

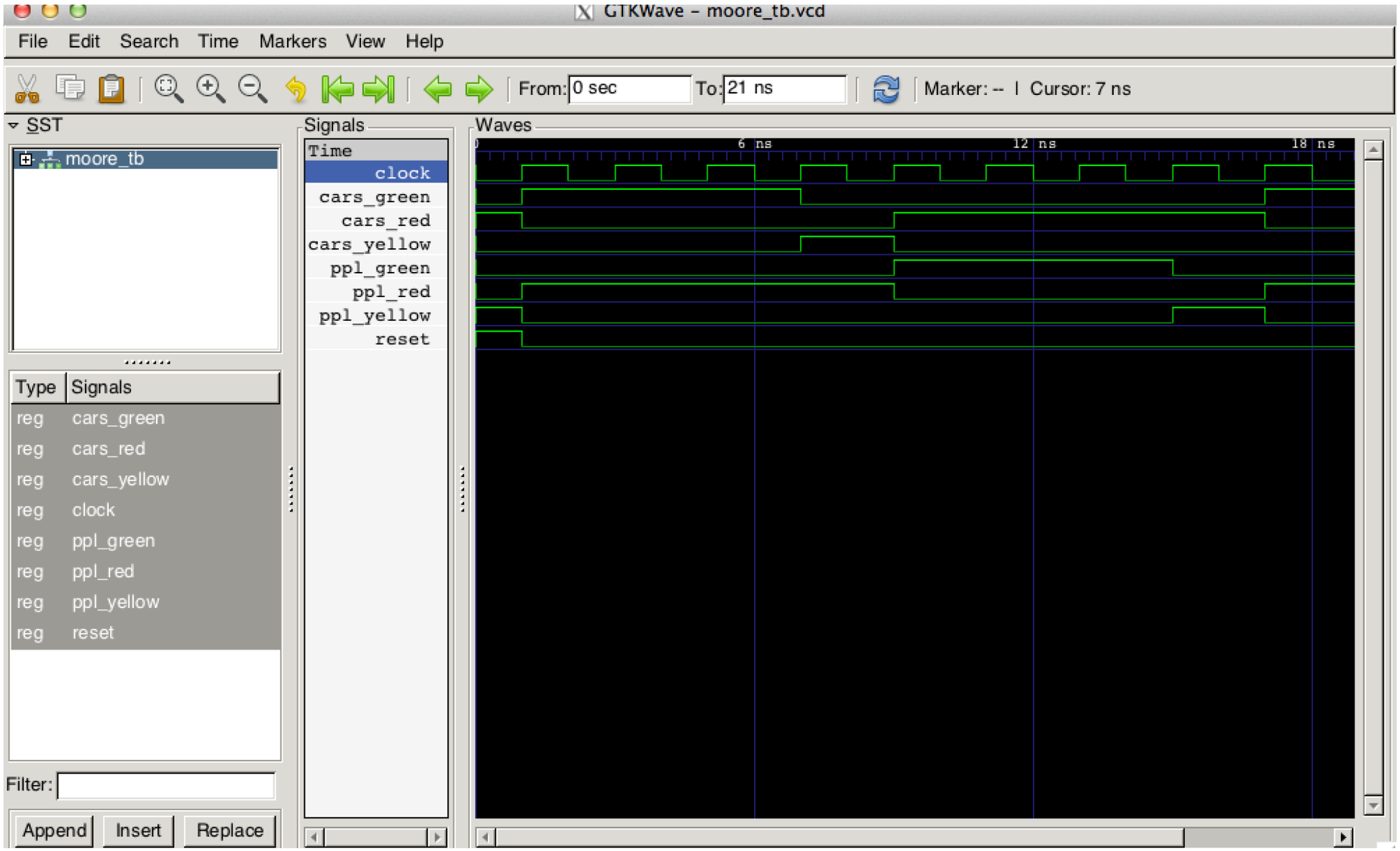
```
defMod.and('clk, 'rst, 'a, 'b, 'z) {
  sync(1).and {
    when ('rst is 1) {
      'z := 0
    } .otherwise {
      'z := 'a && 'b
    }
  }
}
```

Gates and FSM: Gates

- Bitwise operators:
  - &, |, ^, ~
- Logic operators:
  - &&, ||, !
- Others:
  - <<, >>, ~~

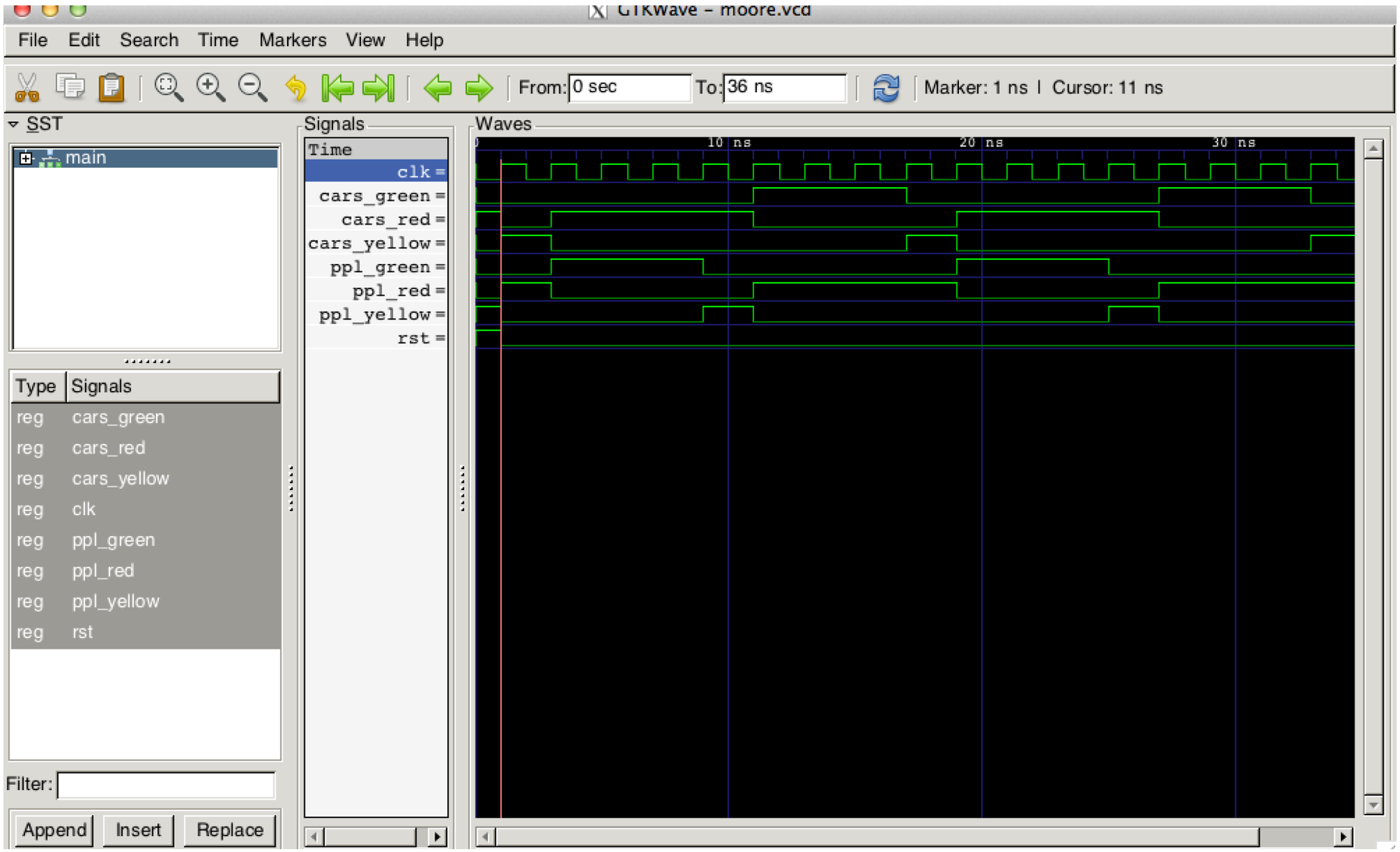
Gates and FSM: FSM

```
when ('rst is 1) {
  timer := 1
  state_traffic := cars_stop.id
} .otherwise {
  when (state_traffic is cars_stop.id) {
    timer := 1
    state_traffic := people_go.id
  } .elsewhen (state_traffic is people_go.id) {
    timer := (timer << 1) % math.pow(2, timerStep1).toInt
    when (timer(timerStep) is 1) {
      state_traffic := people_stop.id
    }
  } .elsewhen (state_traffic is people_stop.id) {
    … }}
```



A Difference with myHDL

This is from myHDL.



A Difference with myHDL

This is from ScalaHDL.

Slicing and Concatenation

- Bit indexing:
  - `‘a(0)`, lsb
- Bit slicing:
  - `‘a(3, 1)` => `a[2: 1]` in Verilog
- Bit concatenation:
  - `‘a ~~ ‘b` => `{a, b}` in Verilog

Unsigned/Signed Types

- assigning a number to a register without enough size to hold it would raise an exception (NotEnoughtBitsException) now

- assigning a negative number to a register of Unsigned type would also raise an exception (IllegalArgumentException) now

Subtraction

```
trait Subtractor extends ScalaHDL {

  val size = 4

  defMod.sub('clk, 'rst, 'a, 'b, 'z) {
    sync(1).sub {
      when ('rst is 1) {
        'z := 0
      } .otherwise {
        'z := ('a - 'b) % math.pow(2, size + 1).toInt
      }
    }
  }
}
```

Modulo

- In Scala:
  - -1 / 4 = 0
  - -1 % 4 = -1
- In Python:
  - -1 / 4 = -1
  - -1 % 4 = 3
- In ScalaHDL, a division/modulo of a Signed register is similar to myHDL.

Signed Type

- If we declare **a** as a 5 bits signed type.

- In Verilog, we will have:

  - input signed [4:0] a;

Operation between Signed and Unsigned Type (Plan)

- ‘a is a 5 bit Signed register.
- ‘b is a 4 bit Unsigned register.
- ‘c is a 6 bit Signed register.
- ‘c := ‘a + ‘b
- In Verilog, we will have:
  - c <= a + $signed({1’b0, b});

Future Plan

- Near future:
  - Implement operations between Unsigned and Signed type.
  - Add some examples to demonstrate slicing, concatenation, etc.
  - Write some more tests.
- Then:
  - ROM and RAM.
  - If statement.

Any Question?