Name1: Abdulrahman Albakkar Name2: Zakaria RAMO 17290643
Id1: 17290662 Id2: 17290643

DESCRIBING THE PROBLEM

Our goal is to describe how a Finite-State machine works using traffic lights as a real-life example in VHDL.

Since a finite state machine gives us the ability of controlling more than one state one after another, we can use it to represent the states of traffic lights we control.

Only one single state of this machine can be active at a time. It means the machine has to transition from one state to another to perform different actions.

In our problem we have four road junctions and in every road junction there is a traffic light and on each one there are three colors (red, yellow, green). Each two roads that are facing each other, like north-south or west-east, is considered to be one, since they behave in the same pattern.

We have 8 states (north-next, start-north, north, stop-north, west-next, start-west, west, stop-west) and in each state there are 6 light signals to be changed (north-red, north-yellow, north-green, west-red, west-yellow, west-green).

The red and green ones take longer than the orange ones. The red and green ones take 60 seconds while the orange ones take 5 seconds. In default we set all the lights signals to be 0. We start from the north-next state that sets only north-red and west-red as 1 for 5 seconds. Next state is start-north: west-red, north-red and north-yellow are set as 1. Next state is north: west-red and north-green are set as 1. Next state is stop-north: only north-yellow and west-red are set as 1. Next state is west-next: north-red and west-red are set as 1. Next state is start-west: only west-red, west-yellow and north-red are set as 1. Next state is west: west-green and north-red are set as 1. Next state stop-north: only north-red and west-yellow are set as 1, And in the next state we go back to the north-next state and continue in this pattern. States that have yellow state = 1 take 5 seconds to proceed, and the other states are considered to take 1 minute to proceed.

WHAT WE LEARNED IN BRIEF

_VHDL is not a software programming language like C or C++, it’s a hardware description language, meaning it is used to describe digital devices.

Name1: Abdulrahman Albakkar Name2: Zakaria RAMO 17290643
Id1: 17290662 Id2: 17290643

If we use a wait statement it will suspend the execution of the process, and the wait for statement will delay the process for a specific time period.

_The difference between a variable and a signal. Variables are good for creating algorithms within a process, but they are not accessible to the outside world.

_std_logic, the std_logic gives us a more fine-grained control over the resources in our design than integer type. Normally, we want a wire in a digital interface to have either the value ‘1’ or ‘0’. These two values are the only values that a big, a binary digit, can have. But in reality, a physical digital signal can be in a number of states, which the std_logic type does a good job emulating. Therefore, it is the most frequently used type in VHDL.

_MODULE, A module is a self-contained unit of VHDL code. Modules communicate with the outside world through the *entity*. *Port map* is the part of the module instantiation where you declare which local signals the module’s inputs and outputs shall be connected to.

Clocked processes are activated if only by master clock signal, not if any other input signal is adapted. The core component of synchronous logic is the flip-flop, there are different forms of it.

Clocked processes are activated at the same time, and the inputs are taken at once, but outputs are taken from the final iteration, which is more efficient for deep logic. The actions in the algorithm can be on clock cycle.

IDEAS TO IMPROVE

Impure functions and procedures could have been used as they improve readability and editability.