Final Project Report

EE 2501 – Digital Logic Design

Contributors:
Joel Metukmebong
Gabriel Rodriguez
Gurbir Singh
Levi J. Vande Kerkhoff

Professor:
Lance Crimm

Due: July 25th, 2023

Table of Contents

Objective

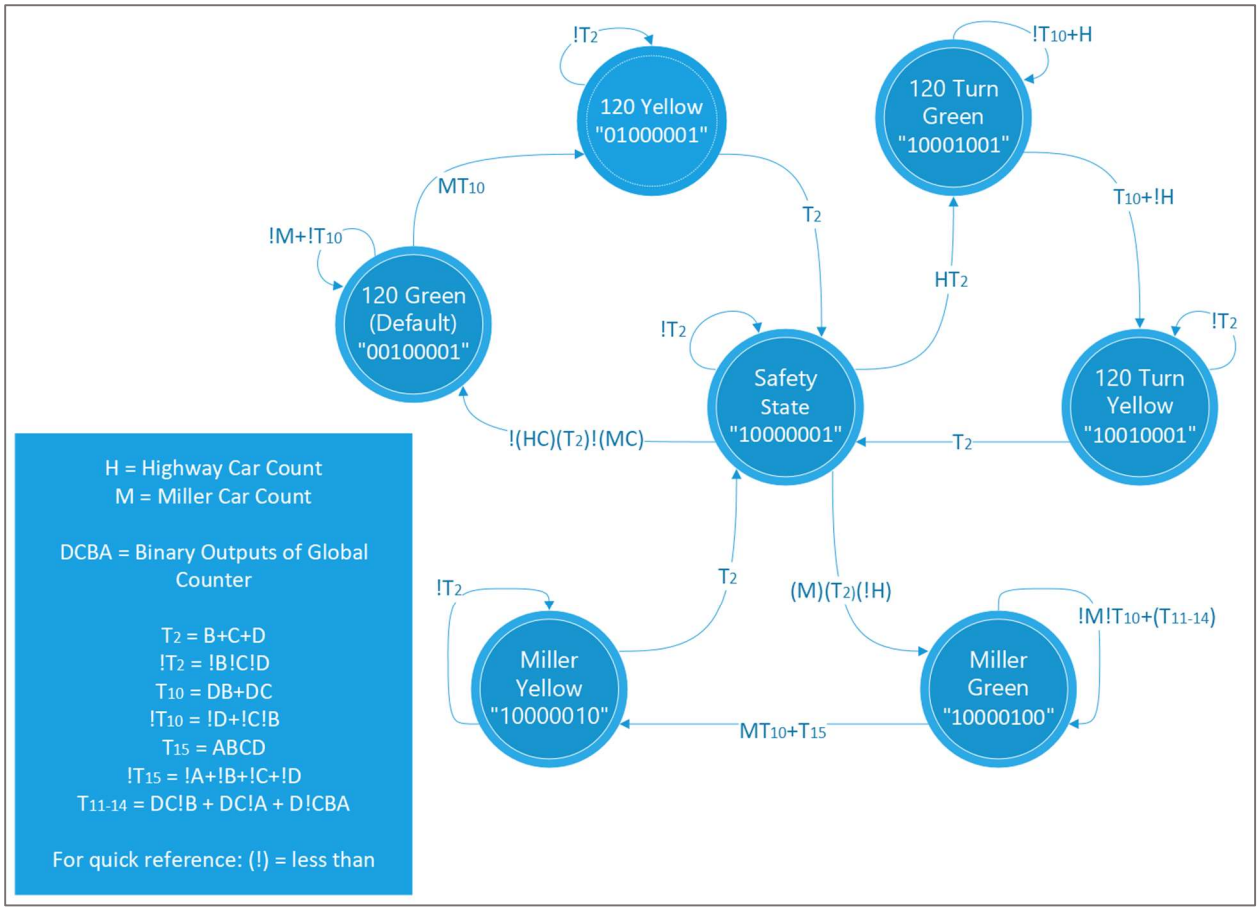
Our final project displays our proficiency in applying state machines, digital logic, and VHDL programming to design and implement a traffic light system for the intersection of Highway 120 and Miller Parkway. The main objective of the project was to create an efficient electronic control system capable of managing the traffic flow at this intersection. The system includes sensors placed to detect the total number of vehicles on both sides of Miller Parkway traveling East or Westbound, and vehicles in the left turn lanes for South-West and North-East traffic. Through this report, we will detail the design, construction, and performance of our traffic light system, highlighting the technical aspects and challenges encountered during its development.

State Machine Design

To design our traffic light system, we began by carefully planning the different states the lights would proceed through and how they would transition between each state. The group first sketched rough ideas across whiteboards and soon transferred the semi-final decisions to the Microsoft Visio diagrams which are shown in the report below.

The finalized state machine starts with the default state where the North-South traffic on Highway 120 has a green light, while the left lanes remain red. The state machine resides in the default state until a sufficient number of vehicles triggers a light change and when the light has been green for 10-seconds at minimum. These vehicles are simulated by two mock vehicle sensors; the turn lane on Highway 120 requires two cars to take priority over default while Miller Parkway requires four. When four or more cars approach either side of Miller Parkway, they are given a green light for at least 10 seconds (but not more than 24 seconds) provided that opposing traffic has already had its minimum green light duration. If the Highway 120 turn lane and Miller

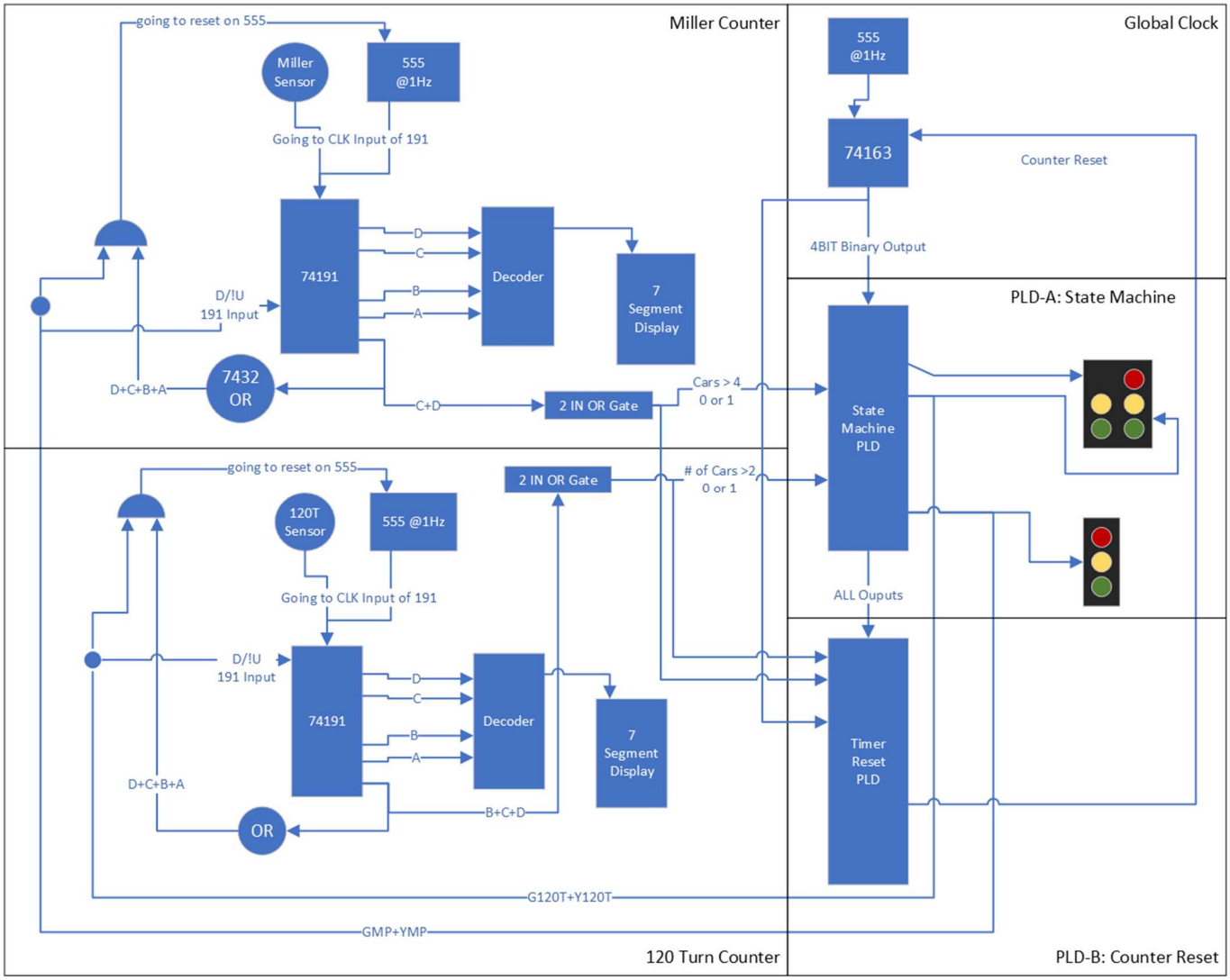
Parkway each reach their respective number of required cars during the default state but before the end of the safety state, then Highway 120 takes priority over Miller Parkway. To ensure all drivers are safe during green-to-red light changes, we included a yellow caution light that activates immediately after the green light turns off and before the red light comes on. This yellow light lasts for two seconds, giving the drivers enough warning. Once each yellow state is complete, the safety state begins. This state lasts for two-seconds and is characterized by both Miller Parkway and Highway 120 being in a red state. A detailed diagram of what has been described can be found below in Figure 1. The 8 binary bits within each state represent the status of lights they correspond to.



(Figure 1: State Machine Diagram)

Circuit diagram

After the design of our state machine was completed, we then started building the circuits based on the diagram below. Our circuit consisted of five sub-circuits: two acting as the sensor for Miller Parkway and the left turn lanes, the state machine (PLD-A), the global clock reset (PLD-B), and the global clock circuit. The sensors were represented using debounced buttons wired to 74191 counters that kept count of the vehicles that had approached. The maximum number of cars that could be counted was restricted to the output of the 74191, thus only a maximum of 15 cars could be counted. The “up-down” controls of the car counter circuits were driven by the outputs of PLD-A; these outputs were the 120 Turn Lane green and yellow outputs and the Miller Parkway green and yellow outputs. During yellow and green states, the respective 555 counters



received high signals on their reset inputs. The 555 clock outputs were wired to the CLK pins of the counters. Each of the three 555 timers were clocked at 1Hz using a combination of 4.7µF capacitors and 33kΩ resistors. This allowed a user to count up each car counter during its respective red-light state and enabled the circuit to emulate cars leaving the intersection during green and yellow states. In order to not count down on the car counters beyond zero and start the counter over at 15, which would defeat the purpose of the emulation, the outputs of the counter were ORed together and ANDed with the aforementioned PLD-A outputs. This combination reset the 555 timer when the counter reached zero. This reset lasted until the state was re-entered. The team felt that this complicated emulation would add complexity and an added challenge to the project.

PLD Design and VHDL Programming

Within our design we utilized two ATMEL 22v10C PLDs: one for the state machine, and the other to reset the global clock counter. PLD-A had 6 inputs and eight outputs. Its outputs controlled the lights directly. The inputs used were the four binary bits from the "global clock" 74163 counter and one bit from each of the car counter subcircuits. As can be seen in the previously discussed state diagram, the PLD-A changed states based solely upon the current state, car counter inputs, and the current time given by the global clock. The outputs of PLD-A and the global clock counter were interpreted by PLD-B using purely combinational logic in order to send a reset signal to the global clock.

Results

The design and construction of the finished circuit required over 100 man-hours in total. The team faced challenges, specifically in the VHDL programming for the PLDs used. In order to comply with changes in logic and continual progression of the circuit, 10+ iterations of the code were implemented and tested during the construction of the circuit. The team also encountered issues with faulty breadboards that led us to believe that our code was incorrect or that other problems had arisen. The breadboards that originally housed both of the car counter subcircuits and had multiple damaged rails. This resulted in wasted time, faulty outputs, and damaged ICs. To solve this problem, the team rebuilt both subcircuits using the breadboards found on two

suitcase power supplies. They proved much more reliable and gave the team no issues. When the individual circuits were finally wired correctly, with corrected logic and VHDL code, a successful finished product was achieved. An improvement that could have been made would be using IR sensors to further emulate the sensing of approaching cars. Another improvement could have been adding one additional counter to each sub-circuit increasing the maximum binary bits to 8 bits. This would have allowed us to count up to 255 cars.

The table below shows a complete list of the components used to complete the project.

Part	Qty	Part	Qty
74LS191N	2	33kΩ Resistor	7
Yellow LED	3	74LS163AN	1
Red LED	2	470Ω Resistor	10
Green LED	3	4.7uF Capacitor	9
74LS47P	3	74LS00	2
2 DIG 7 Segment Display	3	74LS08N	1
Atmel 22v10	2	Push-Button Switch	2
NE555N	3	0.01uF Capacitor	8
7432	2		

Conclusion

In conclusion, through application of knowledge of state machines, digital logic analysis, and VHDL programming the simulation of a traffic light system for the intersection of Highway 120 and Miller Parkway functioned as intended. The four circuits supporting the PLD-A operated in concert with one another and the master PLD creating a functional and integrated solution. While there were challenges with programming and circuity, solutions were found through constantly iterating and improving our code and troubleshooting circuitry.

Appendix A. VHDL Code: PLD-A

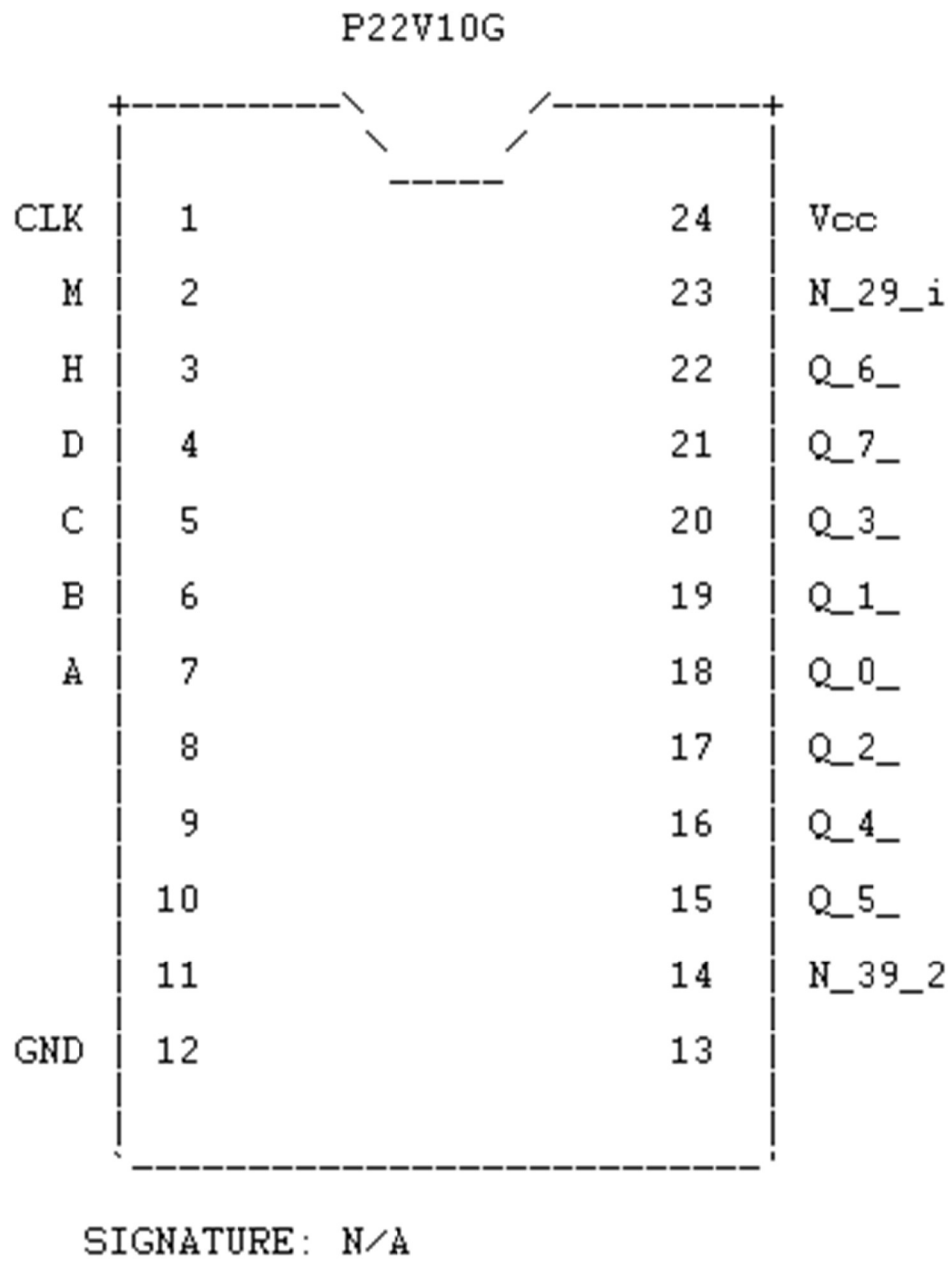
```
library ieee;

use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity Main is
   port (
      CLK:  in    std_logic;
      M:    in    std_logic;
      H:    in    std_logic;
      D:    in    std_logic;
      C:    in    std_logic;
      B:    in    std_logic;
      A:    in    std_logic;
      Q:    buffer      std_logic_vector (7 downto 0));
end Main;

architecture Traffic of Main is
begin
   process(CLK, M, H, D, C, B, A)
   begin
      if CLK = '1' and CLK' event then
         case Q is
         -- 120 Green (Default) ------------------------------------
         when "00100001" => if M = '0' then Q <= "00100001";
            -- (miller cars < 4)
            elsif ((M = '1') or (H = '1')) and (((D and C) or (D and B)) = '1')
then Q <= "01000001"; -- 120 yellow
            -- (miller cars >= 4) and (time > 10)
            end if;

         -- 120 Yellow ---------------------------------------------
         when "01000001" => if ((not B and not C) and not D) = '1' then Q <=
"01000001";
            -- (time < 2)
            elsif ((B or C) or D) = '1' then Q <= "10000001"; -- safety
            -- (time >= 2)
            end if;

         -- Safety State -------------------------------------------
         when "10000001" => if ((not B and not C) and not D) = '1' then Q <=
"10000001";
            -- (time < 2)
            elsif (H = '0') and (((B or C) or D) = '1') and (M = '0') then Q <=
"00100001"; -- 120 green
            -- (highway cars < 2) and (time >= 2)
            elsif (H = '1') and (((B or C) or D) = '1') then Q <= "10001001"; -
- 120 turn green
            -- (highway cars >= 2) and (time >= 2)
            elsif (M = '1') and ((B or C) or D) = '1' and (H = '0') then Q <=
"10000100"; -- miller green
            -- (miller cars >= 4) and (time >= 2) and (highway cars < 2)
```

```
            end if;

         -- 120 Turn Green ----------------------------------------
         when "10001001" => if ((not D or (not C and not B)) = '1') or (H =
'1') then Q <= "10001001";
            -- (time < 10) or (highway cars >= 2)
            elsif (((D and B) or (D and C)) = '1') or (H = '0') then Q <=
"10010001";  -- 120 turn yellow
            -- (time >= 10) or (highway cars < 2)
            end if;

         -- 120 Turn Yellow ---------------------------------------
         when "10010001" => if ((not B and not C) and not D) = '1' then Q <=
"10010001";
            -- (time < 2)
            elsif ((B or C) or D) = '1' then Q <= "10000001"; -- safety
            -- (time >= 2)
            end if;

         -- Miller Green ------------------------------------------
         when "10000100" => if ((not D or (not C and not B)) = '1') or ((M =
'1') and (((D and not C) and (B and not A)) = '1')) or (( ((D and C) and not B)
or ((D and C) and not A) or ((D and not C) and (B and A))) = '1') then Q <=
"10000100";
            -- (time < 10) or (miller cars >= 4)and(time = 10) or (10 < time <
15)
            elsif ((((D and not C) and (B and not A)) = '1') and (M = '0')) or
(((A and B) and (C and D)) = '1') then Q <= "10000010"; -- miller yellow
            -- (time >= 10) and (miller cars < 4) or (time = 15)
            end if;

         -- Miller Yellow -----------------------------------------
         when "10000010" => if ((not B and not C) and not D) = '1' then Q <=
"10000010";
            -- (time < 2)
            elsif ((B or C) or D) = '1' then Q <= "10000001"; -- safety
            -- (time >=2)
            end if;
         -- protection --------------------------------------------
         when others => Q <= "11111111";      -- this is simply for protection

         end case;
      end if;
   end process;
end architecture Traffic;
```

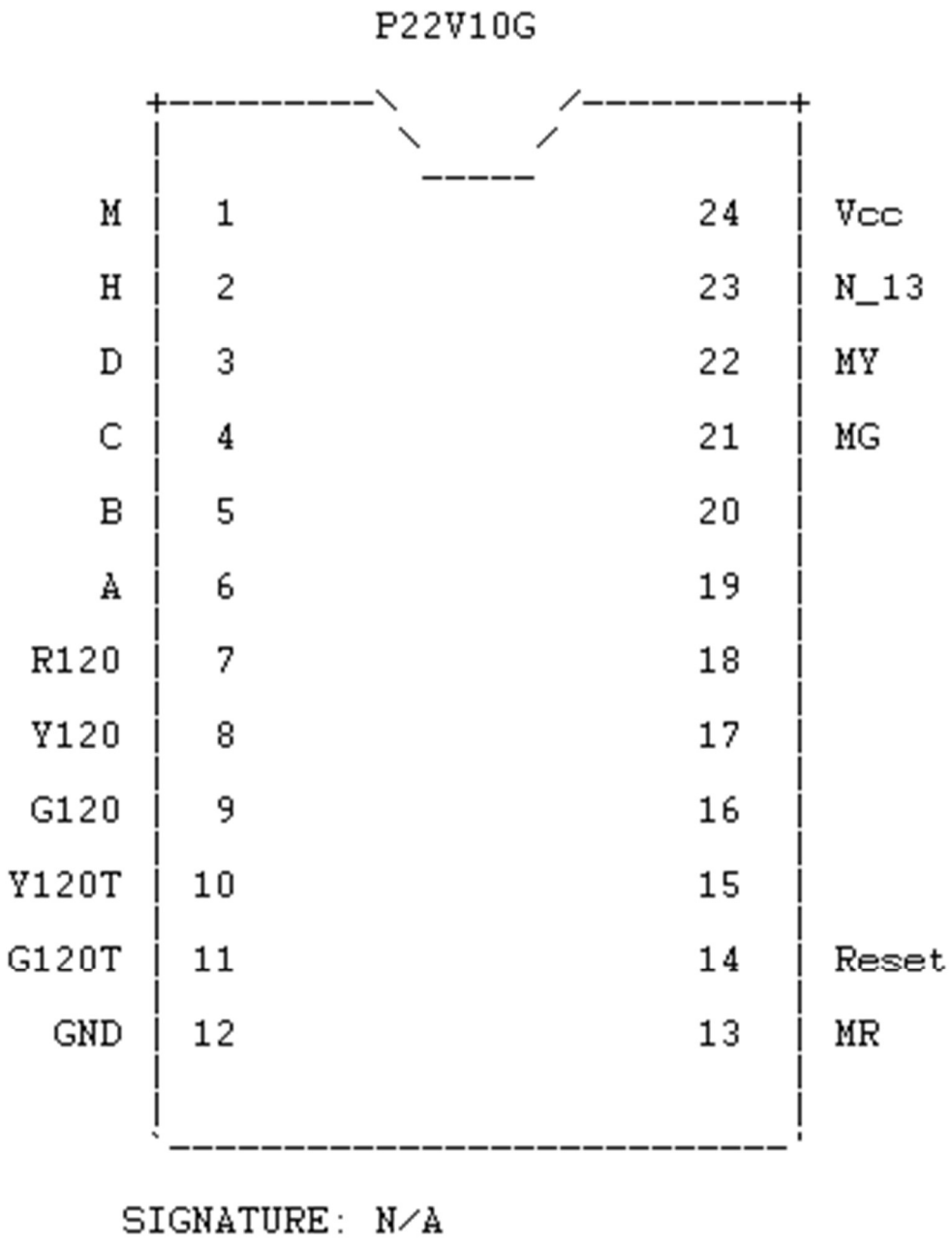
Appendix B. VHDL Code: PLD-B

```
library ieee;

use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity Reset is
   port (
      M:    in    std_logic;
      H:    in    std_logic;

      D:    in    std_logic;
      C:    in    std_logic;
      B:    in    std_logic;
      A:    in    std_logic;

      R120: in    std_logic;
      Y120: in    std_logic;
      G120: in    std_logic;
      Y120T:in    std_logic;
      G120T:in    std_logic;
      MR:   in    std_logic;
      MY:   in    std_logic;
      MG:   in    std_logic;

      Reset:out   std_logic);
end Reset;

architecture Traffic of Reset is
begin
  Reset <=
  ((R120 and MR) and (not Y120T and not G120T) and ((not D and not C) and (B
and not A))) or    -- safety state reset
  ((G120 and (M or H)) and ((D and B) or (D and C))) or
      -- default to 120Y
  (Y120 and ((not D and not C) and (B and not A))) or
      -- 120 is yellow and time is 2
  (G120T and ((D and not C) and (B and not A))) or
      -- 120G turn to 120Y turn
  (Y120T and ((not D and not C) and (B and not A))) or
      -- turn is yellow and time is 2
  ((MG and not M) and ((D and not C) and (B and not A))) or
      -- miller green : change state if miller green and no cars and time is 10
  (MG and ((A and B) and (C and D))) or
   -- miller is green and time is 15
  (MY and ((not D and not C) and (B and not A)));
      -- miller is yellow and time is 2
end Traffic;
```

Appendix C. PLD-A Pin-Out



Appendix D. PLD-B Pin-Out



SIGNATURE: N/A