Automated Gas Station

Presented by :
Malak Mohamed
Mariam Mohamed
Marina Bassem
Hannah Mahmoud
Yassin Ahmed
Omar Khalifa
Yasmine Ahmed

Table of contents

Introduction

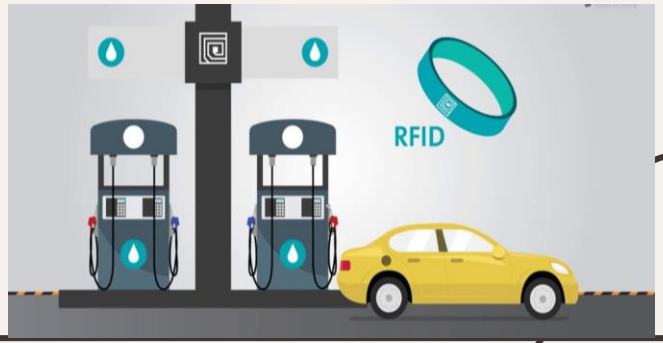
Traditional gas stations have evolved into modern automated facilities as a result of advances in technology and a rise in customer demand for ease of use and effectiveness. These upgraded stations use modern technology to improve customer experience and improve operations, significantly altering how cars are refueled.

Benefits of Automated gas station for costumers

1. Convenience and speed: Automated gas stations are typically available 24/7, allowing customers to refuel their vehicles at any time of day or night without waiting for an attendant. The process is often quicker as customers can directly handle the fueling.
2. Ease of Use: Modern automated gas stations are designed to be user-friendly, with clear instructions and simple payment systems, including credit/debit card readers, mobile payment
3. Reduced Wait Times: With multiple automated pumps, more customers can be served simultaneously, reducing wait times during peak hours.

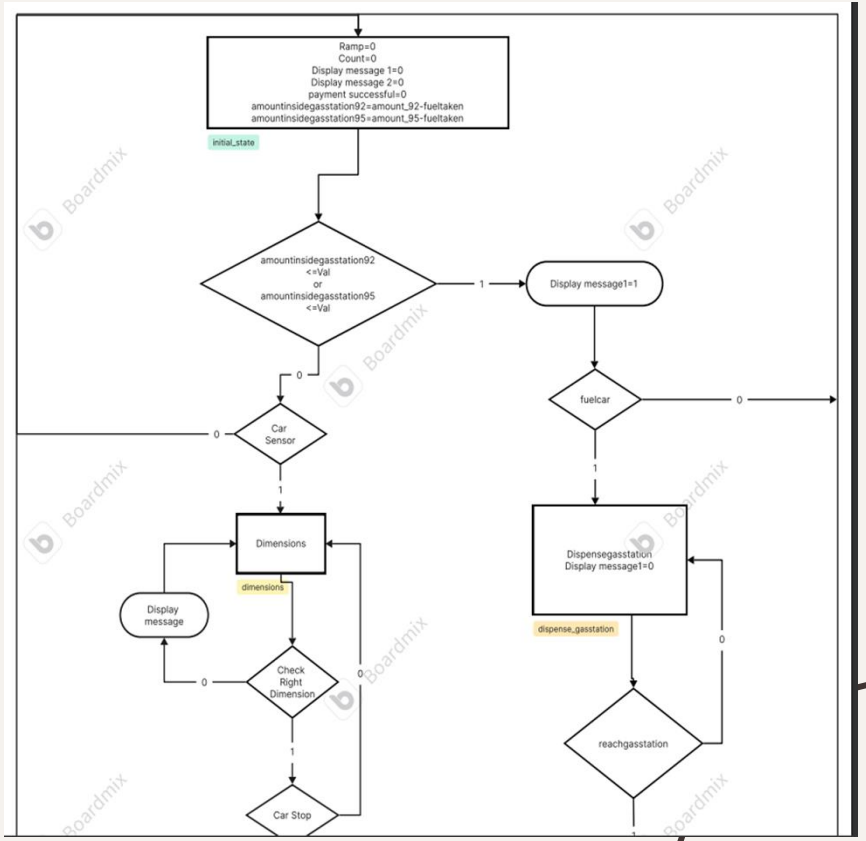
Benefits of Automated gas station for operators

1. Lower Operating Costs: Without the need for attendants, labor costs are significantly reduced. This also minimizes expenses related to employee training, benefits, and payroll management.
2. Extended Operating Hours: Automated stations can operate 24/7 without the need for additional shifts, increasing potential sales and convenience for customers.
3. Increased Efficiency: Automated systems can streamline operations, from fuel delivery to payment processing, reducing errors and improving overall efficiency.
4. Increased Efficiency: Automated systems can streamline operations, from fuel delivery to payment processing, reducing errors and improving overall efficiency.

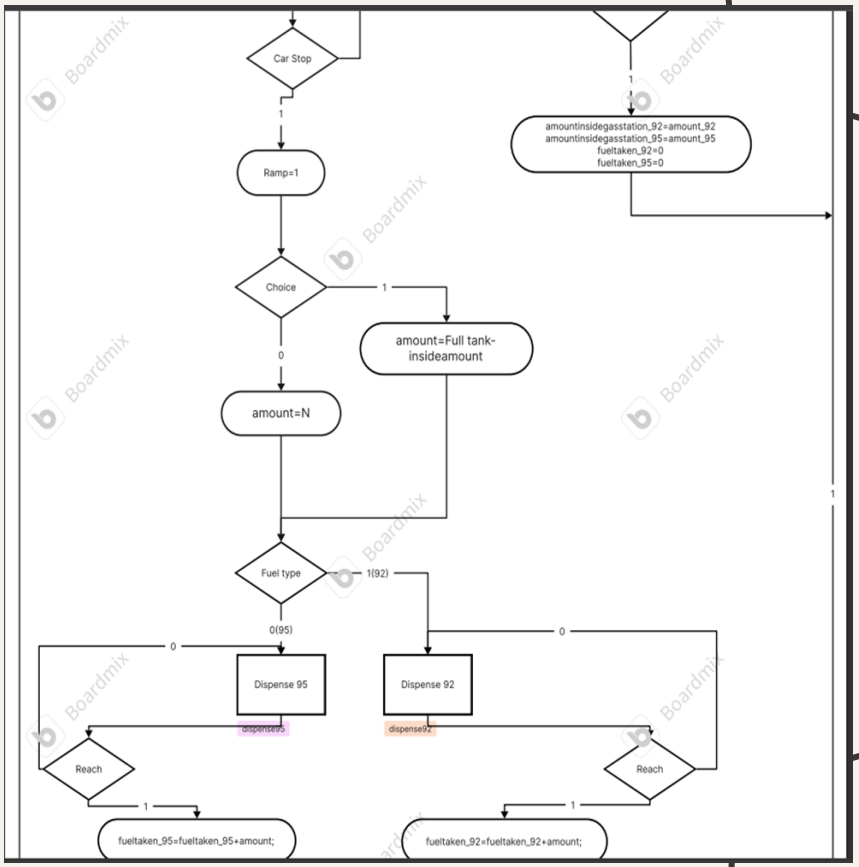


1. Initial state:
   Initialize ramp and counter to 0. Initialize maximum variable. (maximum number of failed tries)
2. Check if amount of fuel 95 or fuel 92 in the gas station is less than sufficient:
   If it is, display a message to the user then proceed to fuel car
3. Fuel car:
   If 0 then return to initial state.
   If 1 then proceed to dispense gas station state and display message.
4. Dispense gas station state
5. Reach gas station:
   If 0 then return to dispense gas station state.
   If 1 then proceed to next statement
6. Statement:
   Amount inside gas station of 92 = amount 92
   Amount inside gas station of 95 = amount 95
   Fuel taken 92 = 0
   Fuel taken 95 = 0
7. Check if amount of fuel 95 or fuel 92 in the gas station is less than sufficient:
   If it is more ,then proceed to car sensor
8. Check if a car is entering the station:
   If the sensor sensed a car entering, proceed to dimension state.
   If it doesn't return to initial state.

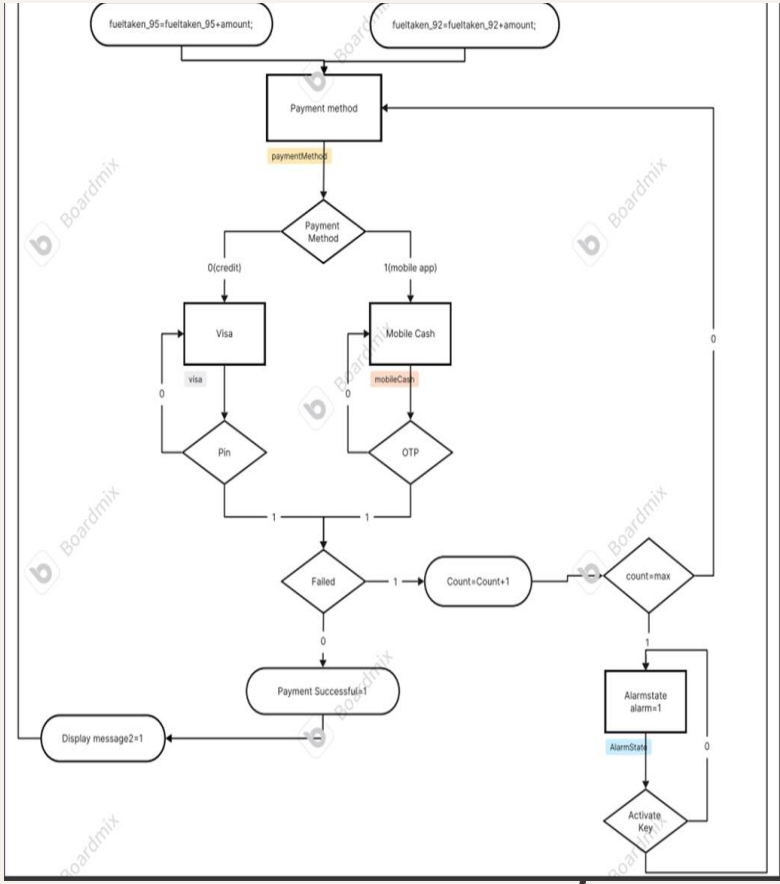
ASM chart



9. Dimension state
10. Check if the car is parked in the right dimensions:
    If it is in the right dimensions proceed to car stop.
    If it doesn't, display a message that tells if to move forward or backward to be in the right dimension
11. Car stops:
    If the car stops then proceed to open the ramp.
    If it doesn't goes back to dimension state.
12. Opens ramp
13. Choice between dispense tank until full or with amount N:
    If 1 then dispense tank until full
    If 0 then enter the amount of fuel needed to be dispensed
14. Fuel Type:
    If 1 then the fuel chosen is fuel 92.
    If 0 then the fuel chosen id fuel 95.
15. Dispense fuel 92 state
16. Reach 92:
    If 0 then return to Dispense 92 state.
    If 1 then add amount to fuel taken 92 then proceed to payment method
17. Dispense fuel 95 state
18. Reach 95:
    If 0 then return to Dispense 95 state. If 1 then add amount to fuel taken 95 then proceed to Payment method.



19. Payment method state
20. Payment method decision:
    If 1 then mobile app.
    If 0 then credit.
21. Mobile Cash (payment method) state
22. OTP:
    If 0 then return to Mobile cash state.
    If 1 then proceed to Failed decision.
23. Visa (Payment method) state
24. Pin:
    If 0 then return to Visa state.
    If 1 then proceed to Failed decision.
25. Failed
    If 1 then start counting how many times the transaction has failed.
    If 0 then payment transaction is successful then return to initial state.
26. Flag decision:
    If 1 then proceed to Alarm state. If 0 then return to payment method state.
27. Alarm state
28. Activate Key decision:
If 0 then return to Alarm state.
If 1 then return to initial state.
29. Back to Initial state again



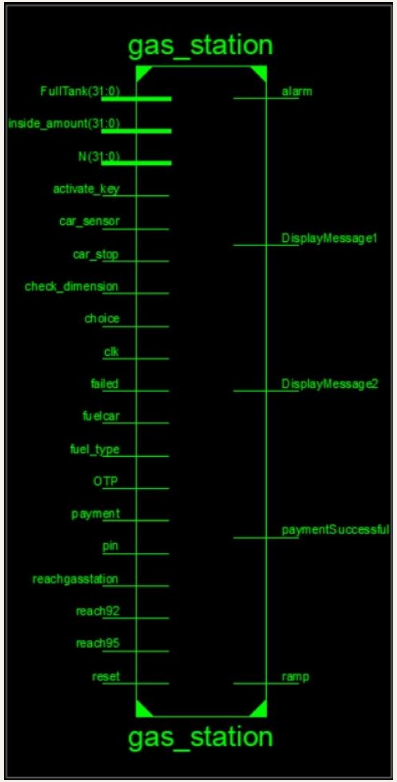
```
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.numeric_std.all;
entity gas_station is
Generic(
max : integer := 3;  -- Maximum value constant
 v: integer := 45;  -- Average liter constant
amount_92:integer:=1000; --maximum amount in the gas station for 92
amount_95:integer:=1000); --maximum amount in the gas station for 95
        port(
clk,reset:in std_logic;
car_sensor,check_dimension,car_stop,choice,fuel_type,reach95,reach92,payment,pin,OTP,reachgasstation,failed,activate_key,fuelcar,flag: in std_logic;
N, FullTank, inside_amount : in integer ;
ramp,alarm,paymentSuccessful,DisplayMessage1,DisplayMessage2: out std_logic  );
end gas_station;
architecture behavioural of gas_station  is
type state is(initial_state, dimensions, dispense95, dispense92,paymentMethod,visa, mobileCash,AlarmState,dispense_gasstation);
signal present_state, next_state:state;
 begin
seq: process(clk, reset)
begin
if reset='1' then
present_state<=initial_state;
elsif rising_edge(clk) then
present_state<=next_state;
end if;
end process seq;
com: process(present_state, car_sensor,check_dimension,car_stop,choice, fuel_type,reach95,reach92,payment,pin,OTP,failed,flag,activate_key,reachgasstation,fuelcar,N, FullTank, inside_amount)
variable amountinsidegasstation_92: integer range 0 to amount_92;
variable amountinsidegasstation_95: integer range 0 to amount_95;
variable count:integer range 0 to 3:=0;
variable fueltaken_92: integer range 0 to amount_92:=0;
variable fueltaken_95: integer range 0 to amount_95:=0;
variable amount : integer ;
begin
case present_state is
when initial_state=>
ramp<='0';
alarm<='0';
DisplayMessage1<='0';
```

```
DisplayMessage1<='0';
DisplayMessage2<='0';
paymentSuccessful<='0';
amountinsidegasstation_92:=amount_92-fueltaken_92;
amountinsidegasstation_95:=amount_95-fueltaken_95;
if (amountinsidegasstation_92<=v) or ( amountinsidegasstation_95<=v)  then
        DisplayMessage1<='1';
        if fuelcar='1' then
                next_state <= dispense_gasstation;
      else
        next_state<=initial_state;
    end if;
elsif car_sensor ='1' then
next_state<=dimensions;
else
next_state<=initial_state;
end if ;
when dimensions =>
if(check_dimension='0')then
report"Please Move" severity note;
next_state<=dimensions;
elsif(car_stop ='1') then
ramp<='1';
else
next_state<=dimensions;
end if;
if(choice='1') then
amount:=FullTank-inside_amount;
else
amount:=N;
end if;
if(fuel_type='1')then
next_state<=dispense92;
else
next_state<=dispense95;
end if;
when dispense95=>
if(reach95='1')then
fueltaken_95:=fueltaken_95+amount;
next_state<=paymentMethod;
else
next_state<=dispense95;
end if;
when dispense92 =>
```

```
when dispense92 =>
if(reach92='1') then
fueltaken_92:=fueltaken_92+amount;
next_state<=paymentMethod;
else
next_state<=dispense92;
end if;
when paymentMethod =>
if(payment='1') then
next_state<=mobileCash;
else
next_state<=visa;
end if;
when visa=>
if(pin='0')then
next_state<=visa;
elsif(failed='1')then
count:=count+1;
if(count=max)then
next_state<=Alarmstate;
else
next_state<=paymentMethod;
end if;
else
paymentSuccessful<='1';
DisplayMessage2<='1';
next_state<=initial_state;
end if;
when mobileCash =>
if(OTP='0')then
next_state<=mobileCash;
elsif(failed='1')then
count:=count+1;
if(count=max)then
next_state<=Alarmstate;
else
next_state<=paymentMethod;
end if;
else
paymentSuccessful<='1';
DisplayMessage2<='1';
next_state<=initial_state;
end if;
when alarmState=>
```

```
next_state<=paymentMethod;
end if;
else
paymentSuccessful<='1';
DisplayMessage2<='1';
next_state<=initial_state;
end if;
when alarmState=>
alarm<='1';
if (activate_key ='1') then
next_state<=initial_state;
else
next_state<=alarmstate;
end if;
when dispense_gasstation =>
DisplayMessage1<='0';
if(reachgasstation='1')then
amountinsidegasstation_92:=amount_92;
amountinsidegasstation_95:=amount_95;
fueltaken_92:=0;
fueltaken_95:=0;
next_state<=initial_state;
else
next_state<=dispense_gasstation;
end if;
end case;
end process com;
end behavioural;
```

Chip design:


gas_station
FullTank(31:0)
inside_amount(31:0)
N(31:0)
activate_key
car_sensor
car_stop
check_dimension
choice
clk
failed
fuelcar
fuel_type
OTP
payment
pin
reachgasstation
reach92
reach95
reset
alarm
DisplayMessage1
DisplayMessage2
paymentSuccessful
ramp
gas_station

Conclusion

In conclusion, automated gas stations mark a significant advancement in the fueling industry. These creative focuses have completely reimagined the conventional gas station experience by utilizing the latest innovations to provide unmatched sustainability, efficiency, and convenience.

The increasing use of automated gas stations has a lot of potential for the future, both for consumers and station operators. The self-service capabilities, continuously accessibility, and advanced features like remote monitoring and alternative fuel choices make these stations well-suited to adapt to the evolving needs of a world that is changing quickly.