# Semi-Automatic Car Anti-Theft Design using ATMega168 Microcontroller

Amevi Acakpovi
Accra Polytechnic (Ghana)
P.o Box: GP561

Quist-Aphetsi Kester
Ghana Technology University College (Ghana)
PMB 100 Tesano

Koudjo Mawuefam Koumadi
University of Ghana
P. o. Box: LG 25.

## ABSTRACT

The design and manufacture of anti-theft systems have become more and more complex due to the rise in complexity of theft in the system. Most of the anti-theft systems available on market, are the alarm types that audibly deter some thieves away but do not prevent one’s car from being stolen and also may not be good enough to meet the growing complexity of theft in the country. This paper presents a simpler and a more efficient car anti-theft system that provides users with improved security by the use of efficient access mechanisms and immobilization systems. Access mechanisms were put in place to request authorization from the user and a hidden button which gives permission to start the car when it is pressed. Immobilizers where put in place to disable car in case it is stolen. The system was based on the ATMega168 microcontroller programming which involve the use electronic components such as limit switches, LEDs and transistors. The programming concept follows the graph theory well known as state machine implementation in Micro-processing and the language adopted was the C programming language.

## General Terms

Automation, Microprocessor/Microcontroller programming, intelligent wireless sensor programming.

## Keywords

microcontroller, car anti theft, security, electronic circuits, state machine.

## 1. INTRODUCTION

Anti-Theft systems have been in existence since individuals began stealing other people's property and have evolved in order to stop increasingly complex methods of theft [1]. The first anti-theft devices to be built were all mechanical devices and were generally locks of various kinds. These have evolved over the years and are still in fashion today. These include steering wheel locks, tire locks, hood locks, gear shift locks, ignition/steering wheel column locks, to mention a few. However, in recent years, crime has become more sophisticated leading to the invention of electronic based car anti-theft systems. They can be divided into three main groups: car alarms, vehicle tracking systems and immobilizers. In this paper we put forward an efficient car anti-theft system with improved features to reduce car-jacking or automobile from being stolen.

## 2. PREVIOUS WORKS

Various types of car alarm system have been in existence for long. A car alarm system is an electronic device or circuit built to alert car owner in case someone attempts to steal the cars. There are many forms of car protection system including vehicle tracking system, car immobilizers and others. Also the notification system varies as technology grows. Commonly encountered car alarm system operation does involve a hidden switch used to activate the alarm system. Automatically the car is off and the hidden switch put in the ON position, the user is given 15 to 30s to get out of the car and close all the doors. After returning and opening the door, a buzzer will sound and the user will be given some 10s to move the switch back to Off position. Failure to do so will make a siren to sound. In other cases, when the alarm system is activated the siren sound as anything comes in contact with the car. In addition to the notification by siren, some recent car alarm systems GSM modems to send SMS or call the user in case of a theft [1]-[3]. Also vehicle tracking system has been adopted to further locate the car even if it is stolen despite all this protective measures.

Another effective car anti-theft system is known as car immobilizer. A car immobilizer is any device which when installed in a car would prevent the car from starting without the authorization of the owner [4]-[5]. It is one of the best ways to prevent thieves from stealing a car even if they gain access to it. In 1993, 144,057 cars were reported to be stolen in Germany. This number was reduced to 57,402 in 2002 after car immobilizers were made compulsory for all new cars in 1998 [6].

## 3. METHOD

This section of the paper describes the steps and procedure involved in the design and construction of a semi- automatic car anti-theft system.

To start with, a state machine was used to explain the main concept of the design. This was followed by the implementation of the state machine in the C programming language using the AVRStudio. The program was compiled and loaded into the microprocessor with adequate circuitry that enabled the physical implementation.

### 3.1 Logical System Implementation

The car anti-theft system will work in four different states namely Idle, Waiting, Alarm and finally Active. To these states, correspond respectively, the digital values 00, 01, 10 and 11. Figure 1 illustrates the state machine.

Initially the system will stay in the idle state (00) and in this state the microcontroller will be in the power saving mode until the user opens the car door. Upon opening the door, the system exits the power saving mode and goes into the waiting state (01). A timer is initiated to give a time limit within which an authorization must be received. Authorization may come in the form of a PIN code being entered, a hidden button being pressed or a phone call being placed. If no authorization is received during the stipulated time, the system changes automatically to the alarm state (10) where the system will activate a siren and call the phone number of the vehicle owner. Also, to prevent the alarm from running down the car battery, the system will return to the waiting state (01) after sometime.

On the other hand, when an authorization is received whilst in the waiting state, the system will go into the active state (11). Once in the active state, the user is allowed to start the car. The system will return to the idle state in three ways. Firstly, it returns if the user opens the door after turning off the key switch. Secondly, it will return if the user presses the distress button even if the key switch is not turned off. Lastly, it will return if a phone call is placed to the GSM modem in the system

## 3.2 Hardware Implementation

The first step in the hardware implementation will consist of assigning to various entities, the corresponding electronic components that best describe their operation.

The distress button is a manual override switch used to trigger the system from the active state to the idle state. When pressed, the microcontroller initiates a countdown timer to arm the system. It could also serve as a means of providing authorization.

A push-to-make switch is the most appropriate for such an application. The car key switch was represented by a Single-Pole Single-Throw (SPST) switch. Also a magnetic field sensor switch was used to determine the opening of the car door. GSM modem serves as both an input and an output entity. A Sony Ericsson K700i mobile phone was used. The siren was represented by a standard 12V car siren. Both the ignition circuitry and fuel pump were represented by 12VDC motors. This allows displaying what happens to those parts of the car when the system engages them.

The final circuit diagram of the system is shown in figure 1.

In the circuit of figure 2, the relays controlling the ignition circuitry, the siren and the key switch have their contacts normally opened while the one controlling the fuel pump is normally closed. Also, the magnetic switch representing the car’s door and the push button representing the distress button are respectively, normally closed and normally opened. The power supply unit as described above supplies the microcontroller with 5V and 12V to the other circuit elements.

# 4. RESULT

The flowchart presented in appendix 1 illustrates the operation of the device and also serves as explanation to the source code shown in appendix 2. The code is written in C programming language under the AVRStudio5 software. The program was compiled and successfully loaded in the ATMega168 microcontroller board. Also the physical implementation including the designing of the power unit, the drive circuit at the output of the microcontroller worked successfully. However, limitations come in failure to make or receive a call due to GSM network imperfections.

# 5. CONCLUSION

In summary this design brings some improved features as compared to existing works. First, the microcontroller allows the system to limit the timing in which authorization must be provided and make the upgradability of the system very flexible. This enabled us to add an alarm system to discourage intruders from stealing the car. Furthermore, the use of a cell phone is incorporated in the system to call the owner of the vehicle, purposely to alert him/her of an intruder. Also, in the worst case where the car is stolen, the phone, which acts as a GSM modem in the car, can make a call to immobilize the vehicle. Finally, a manual override button also immobilizes the vehicle after a certain time when pressed.

This work can be further improved by adopting a modern GSM modem which makes easier, the programming.

# 6. APPENDIXES

## 6.1 Appendix 1: Flowchart Illustrating The Operation of The Device (page 5 and 6)

## 6.2 Appendix 2: Program in C Language

```
/* CODING OF A SEMI AUTOMATIC CAR ANTI THEFT
SYSTEM DATE: 13/02/2013*/

#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/io.h>
#define PortIgnition PORTC
#define BitIgnition PC5
#define DdrIgnition DDRC
#define PortFuel PORTC
#define BitFuel PC4
#define DdrFuel DDRC
#define PortSiren PORTC
#define BitSiren PC3
#define DdrSiren DDRC
#define PinDoor PIND
#define BitDoor PD0
#define PullUpDoor PORTD
#define PinDistressBn PIND
#define BitDistressBn PD1
#define PullUpDistressBn PORTD
#define PinKeySw PIND
#define BitKeySw PD2
#define PullUpKeySw PORTD
#define PortGreenLeds PORTC
#define NbGreenLeds 3
#define DdrGreenLeds DDRC
void MsDelay(unsigned int timing) {
volatile unsigned int i, j;
for (i=0; i<duree; i++) {
for (j=0; j<100; j++);
}
}
void DisplayGreenLeds (char value){
value &= ((1<<NbGreenLeds)-1);
PortGreenLeds &=~((1<<NbGreenLeds)-1);
PortGreenLeds |= value;
}
#define RateSerialIntPD7 8000000/9600/8
#define RateSerialIntPB0 8000000/9600/8
volatile unsigned char bitsSerialIntPD7, cpBitSerialIntPD7;
ISR (PCINT2_vect) {
if (!(PIND&(1<<PIND7))) {
PCICR&=(~(1<<PCIE2));
OCR1A=TCNT1+((RateSerialIntPD7*3)/2);
TIFR1=(1<<OCF1A);
TIMSK1|=(1<<OCIE1A);
cpBitSerialIntPD7=0;
bitsSerialIntPD7=0;
}
}
volatile unsigned char PtRead, PtWrite;
#define lgBuffer 16 unsigned char Buffer[lgBuffer];
ISR (TIMER1_COMPA_vect) {
```

```
cpBitSerialIntPD7++;
OCR1A=TCNT1+RateSerialIntPD7;
if (cpBitSerialIntPD7<=8) {
char val;
bitsSerialIntPD7=(bitsSerialIntPD7>>1)&0x7F;
val=((PIND<<(PIND7-PIND7))&0x80);
bitsSerialIntPD7|=val;
}
else {
if (cpBitSerialIntPD7==8);
else {
TIMSK1&=(~(1<<OCIE1A));
if(((PtWrite+1)%lgBuffer)!=PtRead) {
Buffer[PtWrite]=bitsSerialIntPD7;
PtWrite=(PtWrite+1)%lgBuffer;
}
PCMSK2=(1<<PCINT23);
PCICR|=(1<<PCIE2);
}
}
}
#define DivTimer8 2
#define DivTimer64 3
#define DivTimer DivTimer8
void InitSerial2() {
TCCR1B=(DivTimer<<CS10);
PCMSK2=(1<<PCINT23);
PCICR|=(1<<PCIE2);
PORTD|=(1<<PORTD7);
PtRead=0; PtWrite=0; sei();
TCCR0B=(DivTimer<<CS10);
DDRB|=(1<<PORTB0);
}
unsigned char IfCharSerial2() {
return (PtRead!=PtWrite);
}
unsigned char GetCharSerial2() {
unsigned char car;
while (PtRead==PtWrite);
car=Buffer[PtRead++];
if (PtRead==lgBuffer)
PtRead=0;
return (car);
}
volatile unsigned char carTxCons, cpBitTxCons;
volatile unsigned char signalCarTxCons;
void PutCharSerial2(char c) {
while (signalCarTxCons);
carTxCons=c;
cpBitTxCons=0;
signalCarTxCons=1;
PORTB&=(~(1<<PORTB0));
OCR0A=TCNT0+RateSerialIntPB0;
TIFR0|=(1<<OCF0A);
TIMSK0|=(1<<OCIE0A);
}
ISR (TIMER0_COMPA_vect) {
OCR0A=TCNT0+RateSerialIntPB0;
cpBitTxCons++;
if (cpBitTxCons<=8) {
if (carTxCons&1) PORTB|=(1<<PORTB0);
else PORTB&=(~(1<<PORTB0));
carTxCons=(carTxCons>>1);
}
else {
if (cpBitTxCons==9)
PORTB|=(1<<PORTB0);
else {
if (cpBitTxCons==10) {
TIMSK0&=(~(1<<OCIE0A));
signalCarTxCons=0;
}
}
}
}
void PutString2 (char *ch) {
while (*ch!=0) {
PutCharSerial2 (*ch); ch++;
}
}
enum {Idle, Wait, Alarm, Active};
int State;
char Ring;
void ExecRing () { State=Active; }
void ExecRing2 () { State=Idle; }
void GetCarEchoTel () {
char c;
if (IfCharSerial2()) {
DisplayGreenLeds (0);
c=GetCharSerial2();
switch (Ring) { // detects incoming phone call case 0:
if (c=='R')
Ring++;
break;
case 1:
if (c=='I') Ring++;
else Ring=0;
break;
case 2:
if (c=='N') Ring++;
else Ring=0;
break;
case 3:
if (c=='G') Ring=0;
if (State==Wait || State==Alarm)
ExecRing ();
if (State==Active) ExecRing2();
break;
default: break;
}
}
}
// useful AT commands //---------------------
void SendComAT(unsigned char numero) {
switch (numero) {
case 1:
PutString2("ATD+233209461532;\r");
break;
case 2:
PutString2("ATA\r");
break;
case 3:
PutString2("+++AT\r");
break;
case 4:
PutString2("ATH\r");
break;
case 5:
PutString2("AT+CPAS=1\r");
break;
case 6:
PutString2("+++ATH\r");
break;
break;
}
}
int main () {
InitSerial2();
DdrGreenLeds|= (1<<NbGreenLeds)-1;
/*DdrIgnition|= (1<<BitIgnition); the outputs don't work when
this part is enabled DdrFuel|= (1<<BitFuel); DdrSiren|=
(1<<BitSiren);*/
PullUpDoor|=(1<<BitDoor);
PullUpDistressBn|=(1<<BitDistressBn);
PullUpKeySw|=(1<<BitKeySw);
State = Idle; int Timer =0;
while (1) {
switch (State) {
case Idle: DisplayGreenLeds (0);
PortIgnition&=~(1<<BitIgnition);
PortFuel&=~(1<<BitFuel);
PortSiren&=~(1<<BitSiren);
if (PinDoor&(1<<BitDoor)) {
State = Wait;
Timer = 50;
}
break;
case Wait:
DisplayGreenLeds (1);
PortIgnition&=~(1<<BitIgnition);
PortFuel&=~(1<<BitFuel);
PortSiren&=~(1<<BitSiren);
GetCarEchoTel ();
if (!(PinDistressBn&(1<<BitDistressBn))) {
State = Active;
}
if (Timer==0) {
Timer=20;
State=Alarm;
}
break;
case Alarm:
DisplayGreenLeds (2);
PortSiren|=(1<<BitSiren);
PortFuel|=(1<<BitFuel);
PortIgnition&=~(1<<BitIgnition);
GetCarEchoTel ();
if (Timer >= 17)
SendComAT(1);
if (Timer < 17)
SendComAT(6);
if (!(PinDistressBn&(1<<BitDistressBn))) {
State = Active;
}
if (Timer==0) {
Timer= 50;
State=Wait;
}
break;
case Active:
DisplayGreenLeds (4);
PortIgnition|=(1<<BitIgnition);
PortFuel&=~(1<<BitFuel);
PortSiren&=~(1<<BitSiren);
GetCarEchoTel ();
if (!(PinKeySw&(1<<BitKeySw)) &&
(PinDoor&(1<<BitDoor))) {
MsDelay(1000);
State = Idle;
}
if (!(PinDistressBn&(1<<BitDistressBn))) {
State = Idle;
MsDelay(5000);
}
break;
}
if (Timer>0)
Timer--;
MsDelay(100);
}
}
```

## 7. ACKNOWLEDGEMENT

This work couldn't have been achieved without the help of Brown Nii Akwei Brown, Daniel AGbessi Dzissah and Dorothy N.A.M. Mensah.

## 8. REFERENCES

[1] Mohammed, A., Muntaser,M., Sayel, F. and Suleiman, A.E. 2012. A Practical Design of Anti-Theft Car Protection System Based on Microcontroller. American Journal of Applied Sciences 9 (5): 709-716, 2012.

[2] Muhamad, A.B.K. 2008. Car Security Via SMS. Technical Report. Faculty of Electronic and Computer Engineering. University of Teknikal Malaysia Melaka

[3] David Lane. 2012. Car Alarm System. Carlifornia Polytechnic State University.

[4] Montaser, N.R., Mohammad, A.A., Sharaf A.A. Intelligent Anti-Theft and Tracking System for Automobiles. International Journal of Machine Learning and Computing, Vol. 2, No. 1, February 2012.

[5] Visa, M. I., Asogwa, A. V., Musa, S.Y. GSM Based Anti-theft Security System Using AT&T Command. International Journal of Computational Engineering Research (ijceronline.com) Vol.2 Issue.5, Sept 2012.

[6] Ali. A Design for CAR Anti-Theft System Using Cell Phone.

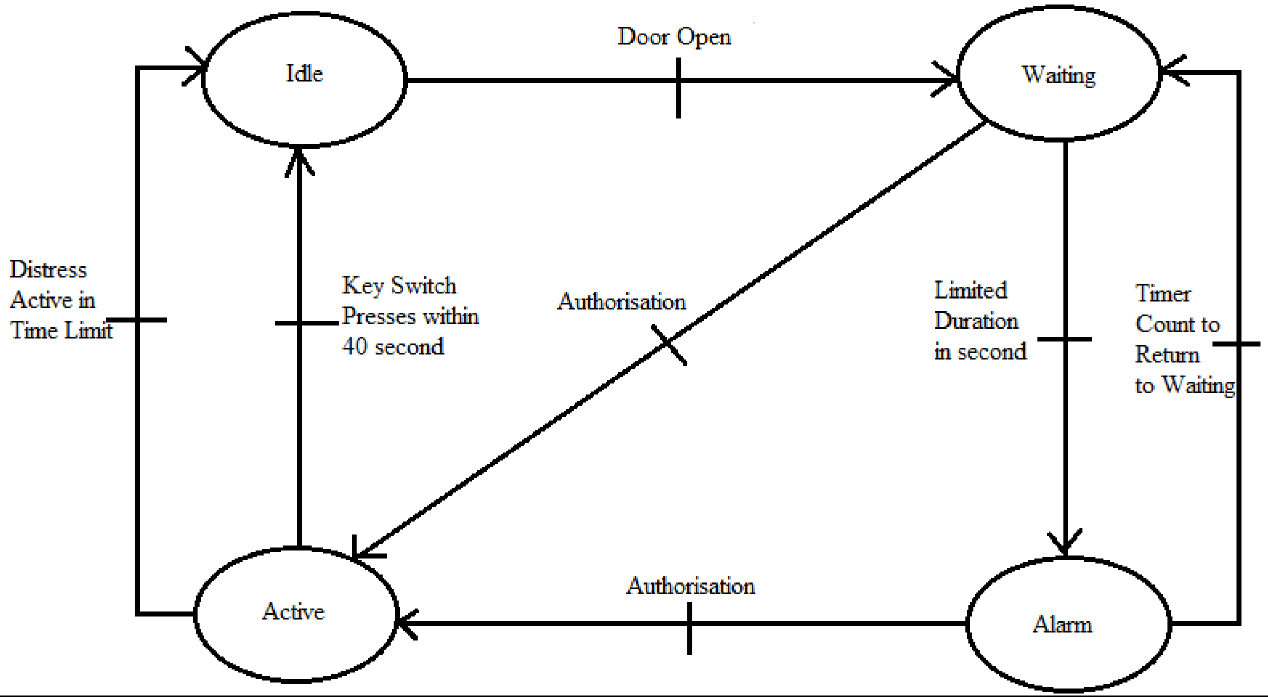
Idle
Door Open
Waiting
Distress Active in Time Limit
Key Switch Presses within 40 second
Authorisation
Limited Duration in second
Timer Count to Return to Waiting
Active
Authorisation
Alarm

**Figure 1: State Machine of the Car Anti-theft System**

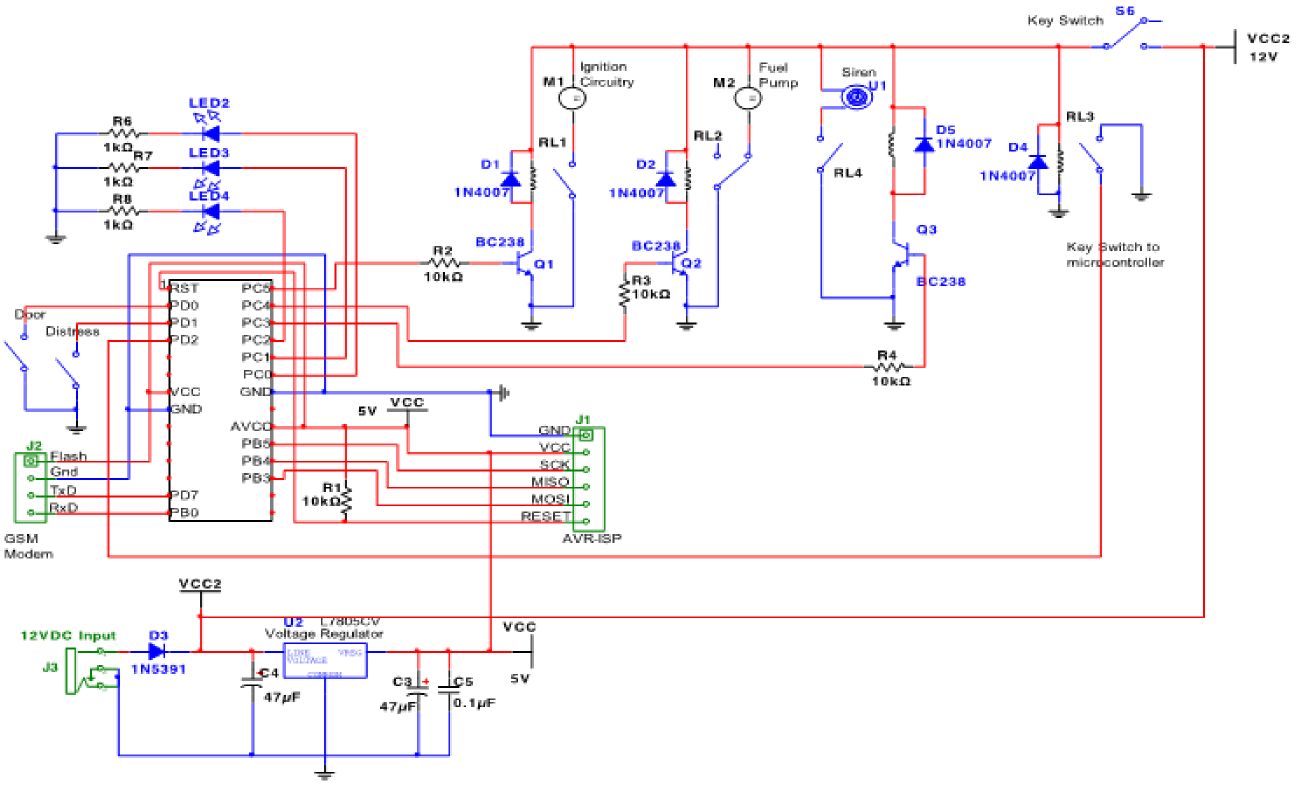

**Figure 2: Complete circuit diagram of the car anti-theft system**

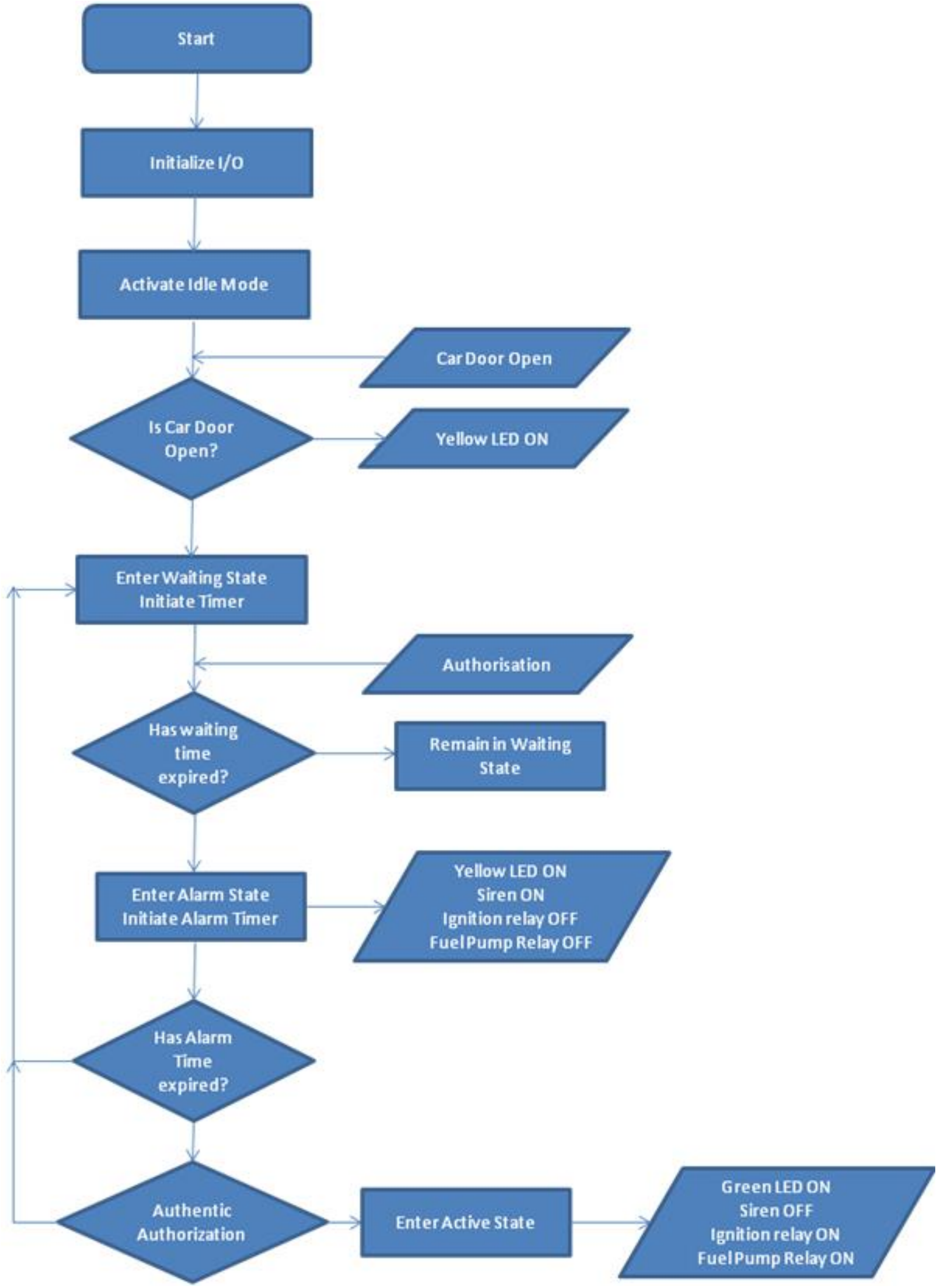

**Figure 3: car anti-theft system Flowchart**