Student: Irimia Petru-Dorin

56SAEA

Proiectarea Unui Circuit de Avertizare Sonoră

Realizarea circuitului s-a efetuat, atât la nivel de simulare cât şi în format fizic. Pentru a proiecta şi simula circuitul, am folosit Proteus unde am realizat circuitul electric, fişierele codului sursă şi proiectarea PCB-ului.

La realizarea circuitului am folosit un microcontroler PIC16F887, 5 butoane, 3 LEDu-ri şi un buzzer. Frecvenţa microcontrolerului este de 8 MHz realizată cu oscilatorul intern. Pentru configurarea butoanelor am setat portul B ca intrare şi citirea acestuia să fie digitală. De asemenea am dezactivat rezistentele pull-up aferente portului B. Butoanele sunt denumite astfel:

- Pentru controlul portierelor DOOR_SENSOR, conectat la RB3;
- Pentru prima poziţie a contactului KLEM15_SENSOR, conectat la RB4;
- Pentru a doua poziţie a contactului KLEM31_SENSOR, conectat la RB5;
- Pentru prezenta centurii de sigurantă BELT_SENSOR, conectat la RB6;
- Pentru starea farurilor LIGHT_SENSOR, conectat la RB7;

Configurarea LED-urilor s-a efectuat prin setarea portului A ca ieşire şi am dezactivat funcţiile analogice.

- LED prezentă portieră deschisă DOOR_LIGHT, conectat la RA0;
- LED avertizare lipsă centură BELT_LIGHT, conectat la RA1;
- LED avertizare lumini LIGHTS_LIGHT, conectat la RA2;

PWM-ului s-a realizat prin configurarea portului RC2 ca ieşire, a perioadei PWM prin setarea lui PR2 (124) la o frecventă de aproximativ 1kHz, activarea modulului PWM prin configurarea registrului CCP1CON şi activarea timerului 2 (T2CON) cu un prescaler de 1:16. Duty cycle – ul este iniţiat prin CCPR1L cu 0. În timpul funcţionarii buzzer-ului acesta este setat la 62 (aprox. 50%).

Formula de calcul pentru frecvenţa PWM este $f_{PWM} = \frac{F_{osc}}{4 \cdot N_{prescaler} \cdot (PR_2+1)}$

F_{osc} = 8 MHz

$N_{prescaler}$ = 16

PR_2 = 124

f_{PWM} = 1000 Hz

Formula pentru Duty-Cycle este $DutyCycle = \frac{CCPR1L \cdot 4 + DC1B}{4 \cdot (PR_2 + 1)} \cdot 100$

CCPR1L = 62

DC1B = 0

PR_2 = 124

DutyCycle = 49.6%

Pentru a se putea genera delay-ul am folosit un timer0 (TMR0) care a fost cofigurat astfel:

- S-a ales un prescaler de 1:8 şi a fost iniţializat TMR0 cu 6 pentru a avea 250 ticks.
- S-a activat întreruperea de timer0 prin activarea bit-ului TMR0IE
- S-au activat întreruperile periferice prin setarea bit-ului PEIE la valoarea 1
- S-au activat întreruperile globale prin activarea bit-ului GIE

Rutina de întreruperi se va activa la fiecare 1ms. În interiorul acesteia se incrementează un counter care este verificat dacă depăşeşte valoarea 10. Dacă această condiţie este indeplinită, variabila folosită in program se va incrementa cu valoarea 10, generând astfel un delay-uri de 10ms.

- Formula pentru delay-ul generat este $T_{delay} = \frac{\text{Pr}\, escaler \cdot (256 - TMR0)}{F_{osc} / 4}$

Logica care stă la baza programului constă in verificarea butoanelor şi tratarea acestora conform cerinţelor:

- Portiera deschisă şi KLEM15(prima poziţie a contactului) activată, vor duce la: generarea unui semnal în care PWM-ul va fi ON 0.5s şi respectiv OFF 0.5s. Acest ciclu se repetă pe toată durata a celor 3s, după care PWM-ul va fi OFF şi LED-ul va rămâne aprins dacă butonul încă este apăsat.
- Centura deconectata şi KLEM15 cat si KLEM31(a doua poziţie a contactului) activate, vor duce la generarea unui semnal în care care PWM-ul va fi ON 0.3s şi respectiv OFF 0.2s. Acest ciclu se repetă pe toata durata a celor 3s, după care PWM-ul va fi OFF. LED-ul va urma şi nega funcţionalitatea de ON şi OFF a semnalului PWM iar la final rămâne aprins dacă butonul incă este apăsat.
- Luminile aprinse şi KLEM15 cât şi KLEM31 deconectate, vor duce la generarea unui semnal în care PWM-ul va fi ON 0.15s şi respectiv OFF 0.1s. Acest ciclu se repetă pe toată durata a celor 3s, după care PWM-ul va fi OFF şi LED-ul va rămâne aprins dacă butonul încă este apăsat.

Astfel, avem trei funcţii de bază care sunt verificate constant. Acestea sunt :

- doorCheck(), cu rolul de a verifica starea portierelor;
- beltCheck(), cu rolul de a verifica starea centurii de sigurantă când maşina e pornită;
- lightCheck(), cu rolul de a verifica starea farurilor în momentul în care maşina nu are alimentare;

În momentul în care există prima poziţie a contactului activă şi o portieră deschisă, se verifică dacă portiera a mai fost deschisă şi în trecut. În cazul în care aceasta nu a fost, se aprinde LED-ul ce indică prezenta unei portiere deschise, se atribuie variabilei ce contorizează delay-ul o valoare de start iar programul intră într-o buclă timp de 3s în care se verifică dacă au trecut 500 ms pentru a oscila semnalul de pe buzzer şi totodată dacă în aceste secunde a fost sau nu dezactivat unul dintre cele 2 butoane. În cazul în care a fost înregistrată şi a doua poziţie a contactului ca fiind adevarată (a fost apăsat butonul), se verifică dacă centura de siguranţă este deconectată şi se va aprinde LED-ul aferent acesteia.

În funcţia beltCheck() este o verificare suplimentară care suspendă execuţia semnalului PWM şi trecerea celor 3s, în cazul în care butonul de la portieră a fost apăsat pentru prima dată. În cazul în care acesta era apăsat în trecut şi se înregistrează o relaxeare a acestuia, LED-ul de stare al portierelor se stinge fară a impacta generarea semnalului PWM.