1. Napisati program za programiranje uC 8051 koji treba da generise povorku impulsa na pinu P1.0. Program napisati koristeci Tajmer 0 koji radi u modu 1. Perioda signala treba da bude u ms tako da impuls I pauza mogu biti razlicitog trajanja. Trajanje impulsa moguce je podesiti u programu. Implementirati u programu signal koji se periodicno ponavlja gde impuls traje 3ms a pauza 2ms pa nakon 5 takvih povorka zameniti trajanje pauze I impulsa za narednih 5 povorka.

Resenje:

#include <reg51.h> // Include 8051 header file

Data unsigned int i;

sbit Test = 0x90; // P1.0

// funkcija za generisanje ms kasnjenja koriscenjem Timer 0 u Mode 1

void delay\_ms (unsigned int k)

{

 // Izracunati vrednost za inicijalizaciju tajmera za i ms

unsigned int timer\_value;

 timer\_value = 65536 - (k \* 1000);

 // konfiguracija Timer 0 u Mode 1 (16-bit tajmer)

 TMOD = 0x01; // Timer 0 Mode 1

 TH0 = timer\_value >> 8; // ucitaj visi bajt u TH0

TL0 = timer\_value & 0xFF; // ucitaj nizi bajt u TL0

 // Start Timer 0

 TR0 = 1;

 // cekaj dok Timer 0 ne bude overflow (TF0 flag = 1)

while (!TF0)

{

}

 // Stop Timer 0

 TR0 = 0;

 // ponisti overflow flag

 TF0 = 0;

}

void main()

{

 // konfigurisi P1 port kao izlazni

 P1 = 0x00; // Initialize P1

Test = 0; // inicijalno postavljanje P1.0 na 0

EA = 1; //globalna dozvola za interapt-e

ET0 =1; //dozvola za tajmerski interapt 0 (kad se desi TF0)

while (1)

{

 For (I=0, I<5, I++)

 {

// generisi impuls

 Test = 1;

 delay\_ms(3); // 3 ms delay

 // Pauza

 Test = 0;

 delay\_ms(2); // 2 ms delay

}

 For (I=0, I<5, I++)

 {

// generisi impuls

 Test = 1;

 delay\_ms(2); // 2 ms delay

 // Pauza

 Test = 0;

 delay\_ms(3); // 3 ms delay

}

 }

 }

1. Kreirati program za 8051 mikrokontroler koji generiše niz impulsa na P1.0 pinu. Period impulsa je inicijalno 1 ms, nakon čega sledi pauza od 1 ms. Dodatno, pomocu tastera 1 povezanog na P3.2 se povećava vreme impulsa dok se pomocu tastera 2, povezanog na P3.3 vreme impulsa smanjuje. ZA realizaciju koristiti Tajmer 0 u režimu 1.

#include <reg51.h> // Include 8051 header file

Data unsigned int ms\_time = 1; //1ms trajanje

sbit Test = 0x90; // P1.0

// funkcija za generisanje ms kasnjenja koriscenjem Timer 0 u Mode 1

void delay\_ms ()

{

 // Izracunati vrednost za inicijalizaciju tajmera za ms\_time ms

unsigned int timer\_value;

 timer\_value = 65536 - (ms\_time \* 1000);

 // konfiguracija Timer 0 u Mode 1 (16-bit tajmer)

 TMOD = 0x01; // Timer 0 Mode 1

 TH0 = timer\_value >> 8; // ucitaj visi bajt u TH0

TL0 = timer\_value & 0xFF; // ucitaj nizi bajt u TL0

 // Start Timer 0

 TR0 = 1;

 // cekaj dok Timer 0 ne bude overflow (TF0 flag = 1)

while (!TF0)

{

}

 // Stop Timer 0

 TR0 = 0;

 // ponisti overflow flag

 TF0 = 0;

}

// interapt servisni potprogram za taster1 (INT0)

void taster1\_ISR() interrupt 0

{

ms\_time += 1;

}

// interapt servisni potprogram za taster2 (INT1)

void taster2\_ISR() interrupt 2

{

if (ms\_time> 1)

{

 ms\_time -= 1;

 }

}

void main()

{

 // konfigurisanje interapta (Button1 and Button2)

 IT0 = 1; // INT0 na ivicu

IT1 = 1; // INT1 na ivicu

EX0 = 1; // Enable INT0

EX1 = 1; // Enable INT1

ET0 =1; //dozvola za tajmerski interapt 0 (kad se desi TF0)

 EA = 1; // Enable globalno interapte

 while (1)

{

 // Generisanje impulsa

 Test = 1;

 delay\_ms(); // cekanje

// Pauza

 Test = 0;

 delay\_ms(); // cekanje

 }

}

1. Napisati program za uC 8051 koji treba da generise povorku impulsa na pinu P1.0. Perioda signala je 300 us tako da impuls traje 150us, pauza 150us. Frekfencija oscilatora je 12 MHz. Program napisati:
2. koristeci Tajmer 1 koji radi u modu 2 bez upotrebe rutine za prekid
3. koristeci Tajmer 0 koji radi u modu 2 upotrebom rutine za prekid

255 - 149 = 106 = 0x6A //vrednost na koju treba napuniti TL1 I TH1

a)

#include <reg51.h> /\* Include x51 header file \*/

sbit pulse = 0x90; /\* P1^0; set test pin0 of port1 \*/

void main()

{

 TMOD = 0x20; /\* Timer1 mode2 (8-bit auto reload timer mode) \*/

/\* ili M0\_0 = 0;

 M0\_1 = 1; ???? but TMOD is not bit addressable; \*/

 TH1 = 0x6A; /\* Load 8-bit in TH1 \*/

 TL1 = 0x6A; /\* Load 8-bit in TL1 once \*/

 TR1 = 1; /\* Start timer1 \*/

pulse = 1; /\* Toggle pulse pin \*/

 while(1)

 {

 while(!TF1); /\* Wait until timer1 flag set \*/

 pulse = !pulse; /\* Toggle pulse pin \*/

TF1 = 0; /\* Clear timer1 flag \*/

 }

}

b)

#include<reg51.h>

sbit pulse = 0x90; /\* P1^0; set test pin0 of port1 \*/

void timer0\_isr() interrupt 1

{

LED =! LED; // Toggle the LED pin,

TF0 = 0;

 //Note Timer value is not reloaded, It is automatically taken care,

// whether the Timer Flag 0 (TF0) needs to be cleared manually or is cleared automatically upon exiting the ISR depends on the specific microcontroller architecture you are using.

For many microcontrollers, especially those in the 8051 family, the TF0 flag is **not** automatically cleared when exiting the ISR. You typically need to clear it manually within the ISR to prevent the interrupt from triggering again immediately after the ISR exits.

U knizi stoji da se TF0 I TF1 automatski hardversi resetuju nakon poziva prekidnog potprograma.

}

void main()

{

TMOD = 0x02; //Timer0 mode 2

TL0 = 0x6A;

 TH0 = 0X6A; //Load the timer value

 TR0 = 1; //turn ON Timer zero

 ET0 = 1; //Enable TImer0 Interrupt

 EA = 1; //Enable Global Interrupt bit

 while(1)

 {

 // Do nothing

 }

}

//In the 8051 microcontroller, when an interrupt occurs, the CPU takes a specific number of machine cycles to jump to the Interrupt Service Routine (ISR) and then back to the main program after the ISR is executed. The exact number of machine cycles can vary depending on the type of interrupt and the instructions used. -

Videti u knizi postoji objasnjenje - minimalan br ciklusa od detektoavanja zahteva za prekidom do izvrsenja prve instrukcije prekidnog potprograma je 3 masinska ciklusa.

Typically, the process involves the following steps:

1. The interrupt is acknowledged during the execution of the current instruction.
2. After the current instruction is completed, the CPU executes an internal “interrupt acknowledge” cycle which takes 2 machine cycles.
3. The CPU then saves the address of the next instruction (where it will return after servicing the interrupt) onto the stack. This operation takes 2 machine cycles.
4. The CPU jumps to the memory location of the ISR, which is a 3-byte operation taking 2 machine cycles.
5. After executing the ISR, a RETI instruction is used to return to the main program. The RETI instruction also takes 2 machine cycles.

So, the minimum number of machine cycles for the CPU to jump to the ISR and back would be **8 machine cycles**. However, this is a simplified scenario. If the ISR contains instructions that require more than 8 bytes of memory space, additional cycles for jumping within the ISR would be needed. Each jump instruction (LJMP) would add 2 more machine cycles to the process.

[It’s important to note that one machine cycle in the 8051 microcontroller consists of 12 oscillator periods1](https://www.rfwireless-world.com/Terminology/8051-instruction-set.html%22%20%5Ct%20%22https%3A//www.bing.com/_blank). Therefore, the actual time taken for these machine cycles will depend on the oscillator frequency of the specific 8051 system you are working with.

1. Napisati program za uC 8051 koji treba da generise povorku impulsa na pinu P1.0. Perioda signala je 2 ms tako da impuls traje 1ms, pauza 1ms. Frekfencija oscilatora je 12 MHz. Program napisati:
2. koristeci Tajmer 0 koji radi u modu 2 bez upotrebe rutine za prekid
3. koristeci Tajmer 0 koji radi u modu 2 upotrebom rutine za prekid

1000:200 = 5; - br cilusa brojanja

255-199 = 56 = 0x38 - vrednost za inicijalizaciju TL0 I TH0

A)

#include <reg51.h> /\* Include x51 header file \*/

sbit pulse = 0x90; /\* P1^0; set test pin0 of port1 \*/

data unsigned int i =0x00;

void main()

{

 TMOD = 0x02; /\* Timer1 mode2 (8-bit auto reload timer mode) \*/

/\* ili M0\_0 = 0;

 M0\_1 = 1; ???? but TMOD is not bit addressable; \*/

 TH0 = 0x38; /\* Load 8-bit in TH1 \*/

 TL0= 0x38; /\* Load 8-bit in TL1 once \*/

 TR1 = 1; /\* Start timer1 \*/

pulse = 1; /\* Toggle pulse pin \*/

 while(1)

 {

 while(!TF1); /\* Wait until timer1 flag set \*/

i++;

If (i>=5)

{

 pulse = !pulse; /\* Toggle pulse pin \*/

 i =0x00;

}

TF1 = 0; /\* Clear timer1 flag \*/

 }

}

B)

#include<reg51.h>

sbit pulse = 0x90; /\* P1^0; set test pin0 of port1 \*/

data unsigned int i =0x00;

void timer0\_isr() interrupt 1

{

i++;

If (i>=5)

{

 pulse = !pulse; /\* Toggle pulse pin \*/

 i =0x00;

}

TF0 = 0;

 //Note Timer value is not reloaded, It is automatically taken care,

}

void main()

{

TMOD = 0x02; //Timer0 mode 2

TL0 = 0x38;

 TH0 = 0X38; //Load the timer value

 TR0 = 1; //turn ON Timer zero

 ET0 = 1; //Enable TImer0 Interrupt

 EA = 1; //Enable Global Interrupt bit

 while(1)

 {

 // Do nothing

 }

}

1. Napisati program za uC 8051 koji treba da broji impulse koji se javljaju na ulaznom pinu P3.2 i meri njihovo trajanje u mikrosekundama. Koristiti Timer 0 u modu 1.

#include <reg52.h>

 5

 6 /\*---------------------------------------------------------

 7 Timer 0 Overflow Interrupt

 8 ---------------------------------------------------------\*/

 9 unsigned int T0\_ISR\_count = 0;

 unsigned int INT0\_ISR\_count = 0; //max 2^16 -1 puta mozemo da restartujemo brojac

unsigned long T0\_pulswidth\_us = 0; // max 2^32 -1 moze biti trajanje impulsa

 10

 11 void T0\_ISR (void) interrupt 1

 12 {

 13 1 T0\_ISR\_count++;

 14 1 TF0 = 0; // Reset the interrupt request

 15 1 }

 16

 17

 18 /\*---------------------------------------------------------

 19 MAIN C function

 20 ---------------------------------------------------------\*/

 21 void main (void)

 22 {

 37 1 /\*--------------------------------------

 38 1 Enable interrupts for timer 0.

 39 1 --------------------------------------\*/

 40 1 ET0 = 1;

 42 1 EA = 1;

 43 1 /\*--------------------------------------

 44 1 Set Timer0 for 16-bit interval timer

 45 1 mode.

 46 1 --------------------------------------\*/

 47 1 TMOD = (TMOD & 0xF0) | 0x09; // Gate0 = 1, C/T =0; M1 = 0, M1 =1

 48 1

 49 1 while (1)

 50 1 {

 51 2 /\*--------------------------------------

 52 2 Clear the timer overflow counter and

 53 2 the timer low and high registers. Then,

 54 2 start the timer.

 55 2 --------------------------------------\*/

56 2

 57 2 TH0 = 0;

 58 2 TL0 = 0;

 59 2

 60 2 TR0 = 1;

 63 2

 64 2 /\*--------------------------------------

 65 2 Wait for the pulse to start.

 66 2 Then, wait for the pulse to end.

 67 2 --------------------------------------\*/

 68 2 while (!INT0);

 69 2 while (INT0);

 INT0\_ISR\_count++;

 71 2 /\*--------------------------------------

 72 2 Compute the width of the pulse -- one

 73 2 clock cycle is 1us for a standard 8051

 74 2 and display it. // if another pulse occurs while we’re calculating this then it can disturb our math

 75 2 --------------------------------------\*/

 76 2 T0\_pulswidth\_us = TH0 \* 0x100; // ucitaj visi bajt

 /\* ili T0\_pulswidth\_us <<8; \*/

T0\_pulswidth\_us = (T0\_pulswidth\_us & 0xFF00) | TL0; // ucitaj nizi bajt

T0\_pulswidth\_us = T0\_pulswidth\_us \* T0\_ISR\_count;

 78 2 }

 79 1 }

1. Na raskrsnici sa semaforom je instaliran IC senzor cija je namena detekcija vozila (pri detektovanju objekta na izlazu senzora je stanje logicka 1). Iz semafora je takodje izveden signal koji je aktivan (logicko stanje je logicka 1) za vreme dok traje zeleno svetlo. Napisati program za za uC 8051 koji treba da registruje broj vozila koja prolaze kroz semafor u toku trajanja zelenog svetla upotrebom Tajmera/brojaca T0.

Signal iz IC senzora treba povezati na pin P3.4 uC8051 dok je signal koji je aktivan (logicko stanje je logicka 1) za vreme dok traje zeleno svetlo doveden na pin P3.2.

#include <reg52.h>

unsigned int vehicle\_count = 0;

void main (void)

{

 TMOD = (TMOD & 0xF0) | 0x0E; // Gate0 = 1, C/T =1; M1 = 1, M0 =0

 while (1)

 {

 TH0 = 0;

 TL0 = 0;

 TR0 = 1;

 while (!INT0);

 while (INT0);

 vehicle\_count = (vehicle\_count & 0x0000) | TL0;

 }

}