URS Međuispit 2021/2022.

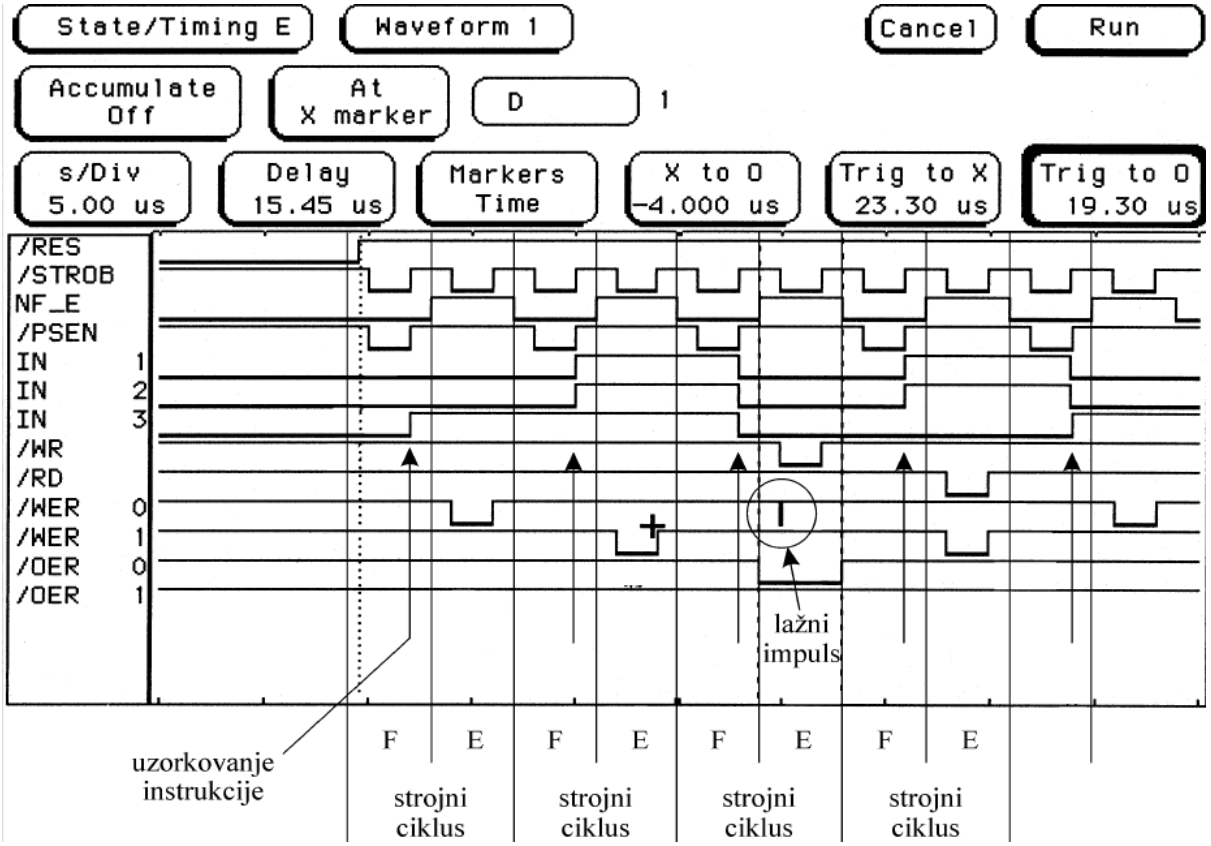
1. zad (1 bod)

Koje sabirnice pomažu procesoru da komunicira s vanjskim svijetom i koje su njihove funkcije? Kojim mehanizmom se može smanjiti broj vanjskih izvoda procesora?

2. zad (3 boda)

(Slika pokazuje rješenje) Pomoću logičkog analizatora snimljeni su vremenski dijagrami na linijama dekodera instrukcije za jednostavni 2-bitni procesor koji je ilustriran na predavanjima. Potrebno je:

a) Označiti strojne cikluse te faze dohvata operacijskog koda i izvršavanje instrukcije
b) Označiti trenutke u kojima se instrukcije uzorkuju u instrukcijskom registru
c) Bez korištenja tablice kodova odgovoriti kojeg su tipa instrukcije koje se izvršavaju
d) Objasniti pojavu „lažnog“ impulsa na liniji /WER0



3. zad (1 bod)

Što su senzori i kako senzore povezujemo s ugradbenim računalnim sustavima. Navedi koje fizikalne veličine je moguće izmjeriti senzorima. Navedi primjere senzora.

4. zad (3 boda)

Na izlaz logičkog sklopa potrebno je spojiti svjetleću diodu koja svijetli kad je na izlazu logička nula. Zadano je izlazni napon VOL = 0.45V uz najveću struju tereta IOL = 40mA. Sklop i LED dioda se napajaju iz izvora napajanja 5V. Skicirati opisani spoj diode. Koji uvjet mora biti ispunjen u ovom spoju i kako to postići? Napon vođenja ove LED iznosi 1.7V, a nominalna struja za puni sjaj iznosi 10mA. Kako ćete ograničit iznos struje ovog spoja?

5. zad (4 boda)

Napiši izraze u sintaksi VHDL-a koji opisuju dvobitni komparator brojeva u dvojnom komplementu. Ulazni vektori neka su X[1..0] i Y[1..0], a izlaz neka je IZ. Za slučaj X>Y, IZ treba postaviti u logičku nulu, a inače u logičku jedinicu. Napiši tablicu istinitosti za sve kombinacije ulaza.

6. zad (2 boda)

Navedi glavne skupine elektroničkih integriranih komponenata obzirom na podjelu po njihovoj funkcionalnosti.

7. zad (2 boda)

Objasnite čemu služi ključna riječ *volatile* i napišite odsječak koda u kojem ćete ilustrirati da je korištenje te ključne riječi nužno za ispravno izvođenje programa.

8. zad (4 boda)

Objasnite algoritam skenirajuće nule za detekciju tipke u matričnoj tipkovnici. Objasnite algoritam na primjeru 4x4 tipkovnice i nacrtajte električnu shemu; za odabrane tipke pokažite na koji način se dobiva kod za njezino očitavanje s porta mikrokontrolera.

9. zad (2 boda)

Objasnite pojam *include guards* i zašto je važan kod *header* datoteka u C-u. Pokažite primjer korištenja *inclued guards* u *header* datotekama.

10. zad (2 boda)

Zaokruži točne tvrdnje

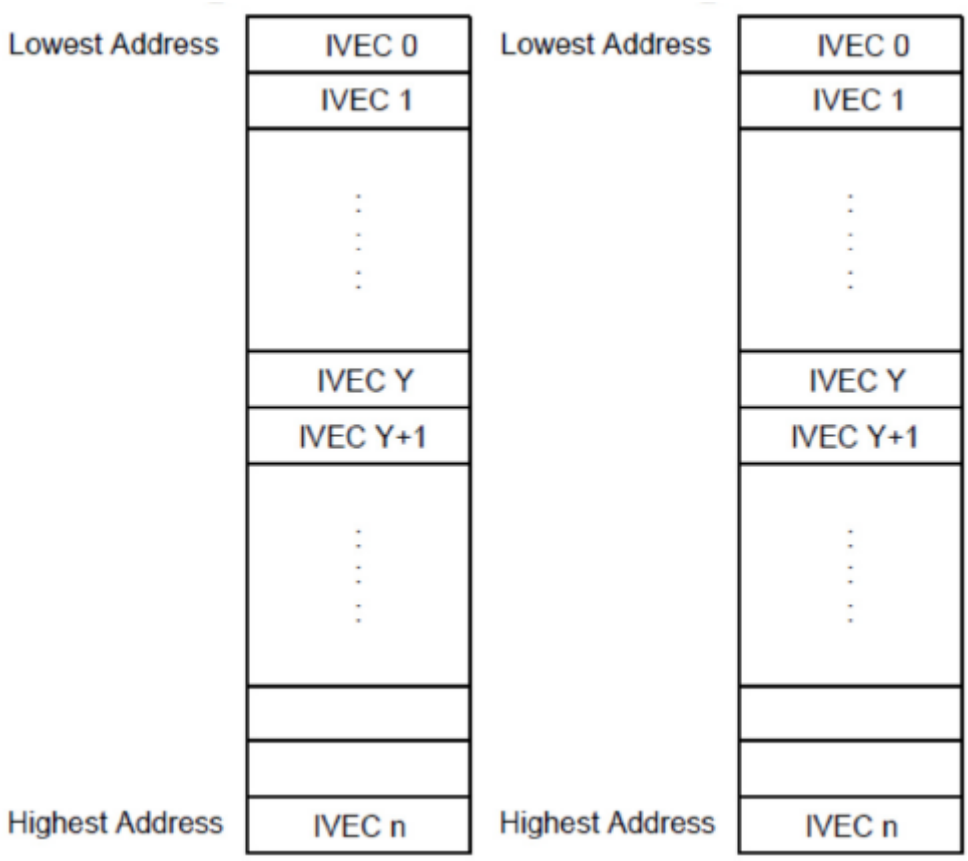
a) Programiranje FUSE bitova (za ATmega4809) moguće je od strane programa koji se izvodi na CPU
b) Registri opće namjene R26-R31 mogu se koristiti na 8 i 16 bitne operacije
c) LOCKBIT fuse trajna onemogućava ponovno programiranje mikrokontrolera putem UDPI sučelja
d) Interni 32kHz ULPO (OSULP32K) služi za implementiranje RTC (Real time counter) za precizno mjerenje vremena.
e) Na ATmega 4809 High Priority (level 1) prekid može prekinuti Normal Priority (level 0) prekide.

11. zad (3 boda)

(Rješenje se nalazi u Prvoj prezentaciji druge cjeline) Objasnite na koji način se mogu promijeniti izvorni prioriteti ATmega 4809. mikrokontrolera. Ako se koristi:

a) Static Scheduling
b) Round Robin scheduling

Označite na slici koji je prekid u prikazanom trenutku najvišeg, a koji najnižeg prioriteta te kako se promatraju prioriteti preostalih prekida u tablici prekidnih vektora ako je CPUINT.CULOPRI=IVEC_Y. Označite mijenja li se automatski i na kojoj slici raspored prioriteta prekida nakon što se izvede trenutni prekid. Objasnite pojam ISR starvation i obrazložite koji od navedenih pristupa rješava taj problem.



12. zad (6 bodova)

Prikazan je dio izvornog koda programa koji koristi TCA0 sklop u *Normal operation* načinu rada za impementaciju vlastite funkcije za realizaciju blokirajuće pauze. Nadopunite nedostajuće dijelove programskog koda tako da program generira pauzu u trajanju od 500ms.

```
/* URS example: TCA_Delay */
#define F_CPU 3333333
#include <avr/io.h>

void delay(uint16_t ms) {
     while(ms--) {




     }
}

void main()
{
uint16_t period;
     // set Normal mode
     TCA0.SINGLE.CTRLB =
     // set period to 1 ms
     period =
     TCA0.SINGLE.PER
     // clear overflow flag
     TCA0.SINGLE.INTFLAGS
     // prescaler x1; enable TCA (run timer)
     TCA0.SINGLE.CTRLA
     PORTD.DIR |
     while(1) {
          delay(500); // blocking delay for 500 ms

     }
}
```

Rješenje

```
/* URS example: TCA_Delay */
#define F_CPU 3333333
#include <avr/io.h>

void delay(uint16_t ms) {
     while(ms--) {
          // wait for OVF
          while((TCA0.SINGLE.INTFLAGS & TCA_SINGLE_OVF_bm) == 0);
          // clear OVF flag
          TCA0.SINGLE.INTFLAGS |= TCA_SINGLE_OVF_bm;
     }
}

void main()
{
uint16_t period;
     // set Normal mode
     TCA0.SINGLE.CTRLB = TCA_SINGLE_WGMODE_NORMAL_gc;
     // set period to 1 ms
     period = (uint32_t)F_CPU / 1000UL;
     TCA0.SINGLE.PER = period;
     // clear overflow flag
     TCA0.SINGLE.INTFLAGS |= TCA_SINGLE_OVF_bm;
     // prescaler x1; enable TCA (run timer)
     TCA0.SINGLE.CTRLA = 0x01;
     PORTD.DIR |= (1 << 7);// set PD.7 as output (LED)
     while(1) {
          delay(500); // blocking delay for 500 ms
          PORTD.OUTTGL = (1 << 7); // toggle LED
     }
}
```