NALOGA N3.1 (2.del) [5T]

Realizirajte "univerzalno" funkcijo ***myUART_Init***, ki lahko nastavi izbrano napravo USART *(npr. USART1, ali USART2,…) in tudi ustrezna priključka TX, RX.* Funkcija prejme parameter tipa ***myUart_TypeDdef***, preko katerega uporabnik poda želje, ki jih naj funkcija nastavi. Uporabite lahko ***privzeto uro procesorja*** (16 Mhz?).

NEOBVEZNO (uporaba funkcij CMSIS): Namesto privzete ure procesorja (konstanta) lahko uporabite tudi sistemsko spremenljivko *SystemCoreClock*, ki vam ob klicu funkcije *SystemCoreClockUpdate* vrne dejansko frekvenco ure procesorja. Tako bo funkcija *myUART_Init* še bolj "univerzalna". 😊

Realizirajte tudi funkcijo ***myUART_Tx***, ki pošlje niz želene dolžine in funkcijo ***myUART_Rx***, ki je namenjena za sprejem znakov. Parameter ***rx_len*** pove, koliko znakov naj funkcija prrebere. Če je rx_len = 0, potem funkcija bere znake do pojavitve znaka ***end_char***. Vse funkcije so ***blokirne*** (uporaba naprave v t. i. načinu "polling"), kar pomeni, da funkcija čaka v zanki na dogodek (npr. postavitev zastavice TXE ali RXNE).

Vaša naloga je, da dopolnite naslednji program, ki nastavi napravo USART2 *(TX=PA2 in RX=PA3, hitrost komunikacije = 115200 b/s)* ter pošlje (prejme) niz. Program vam lahko tudi pomaga pri razumevanju zahtev naloge. Uporabite datoteko ***"stm32f4xx.h",*** kjer so zapisani vsi podatkovni tipi *(npr. GPIO_TypeDef,…),* ki jih najdete v nalogi.

```
#include <stdint.h>
#include "stm32f4xx.h"

#include <stdio.h>
#include <string.h>

typedef struct {
        USART_TypeDef    *USARTx;        //npr. USART1 ali USART2...
        uint32_t          COM_SPEED;     //npr. 115200 ali 9600,...
        GPIO_TypeDef     *TX_PORT;       //npr. GPIOA ali GPIOB, ...
        uint8_t           TX_PIN;        //npr. pin 0 ali pin 1, ....
        GPIO_TypeDef     *RX_PORT;       //npr. GPIOA ali GPIOB, ...
        uint8_t           RX_PIN;        //npr. pin 0 ali pin 1, ....
} myUart_TypeDdef;

void myUART_Init(myUart_TypeDdef *uart_x){

}

void myUART_Tx(USART_TypeDef *USARTx, uint8_t* tx_buff, uint32_t tx_len){

}

void myUART_Rx(USART_TypeDef *USARTx, uint8_t* rx_buff, uint32_t rx_len, uint8_t end_char){

}

uint8_t tx_buf[]= "Test funkcije za pošiljanje\n";
uint8_t rx_buf[100]; //Polje za sprejem zankov

int main(void)
{
        myUart_TypeDdef myUsart2;

        myUsart2.USARTx = USART2;
        myUsart2.COM_SPEED = 115200U;
        myUsart2.TX_PORT = GPIOA;
        myUsart2.TX_PIN = 2;
        myUsart2.RX_PORT = GPIOA;
        myUsart2.RX_PIN = 3;

        myUART_Init(&myUsart2);

        myUART_Tx(USART2,tx_buf,strlen((char*)tx_buf));  //Pošlje niz tx_buff
        //myUART_Rx(USART2, rx_buf, 0, '\n');            //Sprejem znakov se zaključi z znakom '\n'
        //myUART_Rx(USART2, rx_buf, 5, '\n');            //Sprejem znakov se zaključi po 5 znakih

    /* Loop forever */
        for(;;);
}
```

NALOGA N3.2 (2.del) **[5T + 1T + 1T]**

Pri tej nalogi boste prižigali in ugašali diodo LED s pomočjo mobilnega telefona ter pridobili datum in uro iz internetnega časovnega strežnika.

Uporabili boste enega od modulov ***ESPx*** (ESP*8266*/ESP*32*/ESP*32S*) za brezžično komunikacijo Wi-Fi in ga povezali na mikrokrmilnik. Modul boste krmilili prek asinhrone serijske komunikacijo s t. i. ukazi AT. Če potrebujete dostopovno točko (hot-spot), lahko uporabite kar vaš mobilnik.

Za pošiljanje ukazov iz mobilnega telefona si namestite enega od zastonjskih odjemalcev za protocol TCP (npr. TCP client za androida).

Po resetu mikrokrmilnika, se naj le-ta poveže na izbrani časovni strežnik [*NIST Internet Time Servers:* https://tf.nist.gov/tf-cgi/servers.cgi] in pridobi datum in uro.

To Informacijo naj uporabi za nastavitev ure realnega časa (naprava RTC Real Time CLock). **[+1T]**

Ob prejemu ukaza (prvi stolpec v tabeli spodaj) naj izvede ustrezno akcijo (sredinski stolpec v tabeli spodaj) in sporoči akcijo na mobilni telefon v obliki sporočila: “URA.MINUTA: Sporočilo” (desni stolpec v tabeli spodaj). Poskrbite za ustrezno oblikovani izpis na mobilniku. Za nastavitev naprave RTC lahko uporabite tudi grafični vmesnik.

Opomba: Uri realnega časa se lahko izognete, vendar morate v takšnem primeru ob vsakem prejetem ukazu podati zahtevo na internetni časovni strežnik, ki vam vrne aktualni datum in čas. Za vzdrževanje (približnega) aktualnega časa na mikrokrmilniku lahko uporabite tudi običajni časovnik, kateremu se boste podrobneje posvetili pri naslednji vaji (po predčasni potrebi ga preučite in povprašajte [**+1T**, tudi če ste jo že pridobili pri RTC]).

UKAZ IZ MOBILNIKA	**AKCIJA NA MIKROKRMILNIKU**	**POSLANO SPOROČILO DO MOBILNIKA**
“LED ON”	Vključi LED	“12.52: Dioda LED On”
“LED OFF”	Izključi LED	“12.54: Dioda LED Off”
“DATE”	Na novo pridobi trenutni datum in čas ter nastavi RTC.	„12.58: DATUM: 22. 10. 2023, URA: 11:52“
Karkoli drugega		„???“

Ob pritisku na modro uporabniško tipko na razvojni plošči naj aplikacija izvede isto akcijo, kot ob prejemu ukaza “DATE”. Za preverjanje tipke uporabiti prekinitveno strežno rutino.

Pri uporabi modula si pomagajte s priloženim dokumentom (*ESP8266_AT_Examples_v0_4.pdf. Z*a hitri začetek, pa lahko osnove uporabe tudi demonstriram. Za testiranje delovanja samega modula je dobrodošel program za serijsko komunikacijo, kot je npr. Hercules Setup Utility (omogoča tudi nastavitev odjemalca in strežnika za TCP, kot tudi UDP). Lahko ga torej uporabite tudi za testiranje delovanja časovnega strežnika (jeziček TCP Client).