2. domača naloga

MORSEJEV ODDAJNIK

Mentor: Avtor:

Rozman Robert Jožef Poklukar

Vsebina

[1 Delovanje programa 3](#_Toc157450441)

[1.1 Začetne spremenljivke in konstante 3](#_Toc157450442)

[1.2 Podprogrami 3](#_Toc157450443)

[1.2.1 XMCHAR 3](#_Toc157450444)

[1.2.2 XMCODE 3](#_Toc157450445)

[1.2.3 GETMCODE 4](#_Toc157450446)

[1.2.4 XWORD 4](#_Toc157450447)

[1.2.5 morse\_blink 4](#_Toc157450448)

[1.3 Glavni program 4](#_Toc157450449)

[1.3.1 main 4](#_Toc157450450)

[1.3.2 loop 4](#_Toc157450451)

[2 Delovanja dodatnega dela naloge 5](#_Toc157450452)

[2.1 Cilj 5](#_Toc157450453)

[2.2 Konstante 5](#_Toc157450454)

[2.3 Inicializacija gumba 5](#_Toc157450455)

[2.4 Preverjanje pritiska na gumb 5](#_Toc157450456)

[3 Spremenjeni in dodani deli programa 6](#_Toc157450457)

# Delovanje programa

## Začetne spremenljivke in konstante

Prva konstanta A je za pridobitev ascii kode velike črke a, ki jo rabim za pridobitev zaporedne željene črke abecede.

Nato je pod konstanto MORSE sama morsejeva abeceda, ter na koncu spremeljivka NIZ, ki zaseda prostor, kamor se shranjuje vpis.

## Podprogrami

Najprej so programi, ki smo jih delali na vajah:

INIT\_IO:

Vklopi izhodno napravo, v tem primeru zeleno led diodo. Na vajah smo vklapljali rdečo diodo, v tej nalogi sem za vajo in razumevanje naredil enako za zeleno.

LED\_ON:

Pridobi GPIOJ\_BASE naslov ter vanj z odmikom GPIOx\_BSRR vpiše bite 0x00040000, kar vklopi led diodo.

LED\_OFF:

Pridobi GPIOJ\_BASE naslov ter vanj z odmikom GPIOx\_BSRR vpiše bite 0x00000004, kar vklopi led diodo.

INIT\_TC:

Inicializira notranji časovnik, ki si ne bo delil procesorskega časa, zaradi česar je bolj natančen.

DELAY\_TC:

Prebere vrednost iz registra r0, ter vklopi časovnik za prebrano število milisekund.

INIT\_USART3:

Inicializira serijsko komunkacijsko napravo.

SEND\_UART in RECV\_UART:

Prvi pošlje želeni znak iz r0 povezani napravi, drugi pa pridobi znak ter ga shrani v r0.

### XMCHAR

Simbol v r0, ki je '.' ali '-', primerja s '.'. Če sta ascii kodi enaki potem se led dioda prižge za 150 ms, sicer pa za 300.

### XMCODE

V r0 dobi naslov, kjer so shranjeno zaporedje pik in pomišljajev. Za vsak znak pokliče XMCHAR, dokler ne naleti na 0.

### GETMCODE

Ascii kodi črke v r0 odšteje ascii kodo 'A', tako pridobi katera zaporedna črka abecede je bila v r0. Dobljeno število pomnoži s šest, ker je morsejeva abeceda v pomnilniku zapisana po 6 bytov na črko. Dobljen zmnožek in naslov MORSE konstante sešteje, shrani v r0, ter se vrne od koder je bila klicana.

### XWORD

V r0 dobi naslov, kjer so shranjene zaporedne črke. Za vsako pokliče GETMCODE, ter nato XMCODE, ki poskrbita da se željena črka prikaže na izhodni napravi. XWORD se konča, ko naleti na 0.

### morse\_blink

pridobi naslov NIZa, kjer je shranjen željen izhod. Preden pokliče XWORD počaka sekundo, sicer dioda začne izpisovati led diodo preden jo lahko pogledaš.

## Glavni program

### main

Kliče inicializacijske podprograme INIT\_IO, INIT\_TC in INIT\_USART3. Izklopi led diodo in počaka pol sekundo, sicer se prvič, ko se led dioda prižge zmeraj izgleda kot da je bil znak '-'.

### loop

Čaka na vhode iz strani računalnika. Če je vhod »enter« potem takoj kliče more\_blink, sicer pa čaka na 8 vpisanih znakov.

# Delovanja dodatnega dela naloge

## Cilj

Za dodatno nalogo sem želel uporabiti neko vhodno napravo, ki je nismo uporabili pri vajah ali predavanjih. Zastavil sem si, da bom dodal funkcijo modremu uporaniškemu gumbu, ki je na plošči in bo kot ob vnosu tipke enter na računalniku, zažmigal že vnešene črke.

## Konstante

V začetni del programa sem h konstantam dodal tri stvari, in sicer USER\_BUTTON\_PIN, ki pove, na kateri pin je vezana vhodna naprava, v mojem primeru modri uporabniški gumb, GPIOC\_BASE, ki je naslov vmsenika, ter GPIOx\_IDR, ki pa je odmik vhodnega registra (Input Data Register).

## Inicializacija gumba

Nastavitve gumba sem inicializiral v podprogramu INIT\_USER\_BUTTON. Ta zažene uro GPIOC, v MODER nastavi bita 26 in 27 na 00, kar pomeni, da bo pin 13 v načinu vhoda. Na koncu podprogram še resetira IDR register, saj so njegove vrednosti na začetku nastvljene na 1.

Inicializacijo sem dodal k ostalim v začetek glavnega programa.

## Preverjanje pritiska na gumb

Program preverja, če je uporabnik pritisnil na gumb vsakič v zanki RECV\_LP, katera čaka na vhod iz računalnika. Zanka kliče podprogram CHECK\_USER\_BUTTON, v katerem se primerja 13 bit iz IDR registra v GPIOC. Če je ta nastavljen na 1, potem je bil gumb pritisnjen in pokliče morse\_blink.

# Spremenjeni in dodani deli programa

Spodaj navajam dele programa, ki sem jih spremenil oziroma dodal:

// Register Addresses

.equ DWT\_CYCCNT, 0xE0001004 // DWT\_CYCCNT reg (RM, pp.3211)

// RCC base address is 0x58024400

// AHB4ENR register offset is 0xE0

.equ RCC\_AHB4ENR, 0x580244E0 // RCC AHB4 peripheral clock reg

.equ GPIOJ\_BASE, 0x58022400 // GPIOJ base address)

.equ GPIOx\_MODER, 0x00 // GPIOx port mode register

.equ GPIOx\_ODR, 0x14 // GPIOx output data register

.equ GPIOx\_BSRR, 0x18 // GPIOx port set/reset register

// Values for BSSR register - pin 2: LED is on, when GPIO is off

.equ LEDs\_OFF, 0x00000004 // Setting pin to 1 -> LED is off

.equ LEDs\_ON, 0x00040000 // Setting pin to 0 -> LED is on

//blue user button

.equ USER\_BUTTON\_PIN, 13 // user button pin

.equ GPIOC\_BASE, 0x58020800 // GPIOC base address

.equ GPIOx\_IDR, 0x10 // GPIOx input register

// Start of data section

.data

.align

A: .ascii "A"

MORSE: .ascii ".-"

.byte 0,0,0,0 //A

.ascii "-..."

.byte 0,0 //B

.ascii "-.-."

.byte 0,0 //C

.ascii "-.."

.byte 0,0,0 //D

.ascii "."

.byte 0,0,0,0,0 //E

.ascii "..-."

.byte 0,0 //F

.ascii "--."

.byte 0,0,0 //G

.ascii "...."

.byte 0,0 //H

.ascii ".."

.byte 0,0,0,0 //I

.ascii ".---"

.byte 0,0 //J

.ascii "-.-"

.byte 0,0,0 //K

.ascii ".-.."

.byte 0,0 //L

.ascii "--"

.byte 0,0,0,0 //M

.ascii "-."

.byte 0,0,0,0 //N

.ascii "---"

.byte 0,0,0 //O

.ascii ".--."

.byte 0,0 //P

.ascii "--.-"

.byte 0,0 //Q

.ascii ".-."

.byte 0,0,0 //R

.ascii "..."

.byte 0,0,0 //S

.ascii "-"

.byte 0,0,0,0,0 //T

.ascii "..-"

.byte 0,0,0 //U

.ascii "...-"

.byte 0,0 //V

.ascii ".--"

.byte 0,0,0 //W

.ascii "-..-"

.byte 0,0 //X

.ascii "-.--"

.byte 0,0 //Y

.ascii "--.."

.byte 0,0 //Z

NIZ: .byte 0,0,0,0,0,0,0,0,0 // 9 praznih rezerviranih za input (ena dodatna, da se funkcija ustavi, ko naleti na 0)

// Start of text section

.text

.type main, %function

.global main

.align

main:

bl INIT\_IO // Priprava za kontrolo LED diode

bl INIT\_TC // Priprava SysTick časovnika

bl INIT\_USER\_BUTTON // Priprava user gumba

bl INIT\_USART3 // Priprava USART3 naprave

bl LED\_OFF

mov r0, #500

bl DELAYTC

mov r1, #9 //števec za dolžino besede

ldr r2, =NIZ //Kazalec na naslov kamor shranjuje črke

//zanka pridobiva znake iz računalnika in jih hkrati pošilja nazaj, da uporabnik vidi kar je vpisal

loop:

subs r1, r1, #1

beq morse\_blink

bl RECV\_UART

cmp r0, #13

beq morse\_blink

bl SEND\_UART

strb r0, [r2]

add r2, r2, #1

b loop

\_\_end: b \_\_end

// inicializacija user gumba

INIT\_USER\_BUTTON:

push {r0, r1, r2, lr}

ldr r1, =RCC\_AHB4ENR

ldr r0, [r1]

orr r0, r0, #(1 << 2) // Set bit 2 to enable GPIOC clock

str r0, [r1]

ldr r1, =GPIOC\_BASE

ldr r0, [r1, #GPIOx\_MODER]

mov r2, #0xF3FFFFFF // Maska za nastavitev na input

and r0, r0, r2

str r0, [r1, #GPIOx\_MODER] // Nastavi na input v MODER

mov r2, #0x10000000

str r2, [r1,#GPIOx\_BSRR] // nastavi IDR register na 0

pop {r0, r1, r2, pc}

// funkcija, ki preveri, če je bil user gumb pritisnjen

CHECK\_USER\_BUTTON:

push {r0, r1, lr}

ldr r1, =GPIOC\_BASE

ldr r0, [r1, #GPIOx\_IDR]

and r0, r0, #(1 << USER\_BUTTON\_PIN)

cmp r0, #(1 << USER\_BUTTON\_PIN) // Naloži IDR register in pogled 13-i pin, če je na 1

beq morse\_blink

pop {r0, r1, pc}

//Pridobivanje znakov

RECV\_UART:

push {r1, r2, lr}

ldr r1, =USART3\_BASE

RECV\_LP:

bl CHECK\_USER\_BUTTON

ldr r2, [r1, #USART\_ISR]

tst r2, #(1 << 5) // RXNE flag

beq RECV\_LP

ldr r0, [r1, #USART\_RDR]

pop {r1, r2, pc}

//Ko je uporabnik pritisnil enter ali pa vnesel 8 znakov, ta podprogram resetira r0 števec na prvo //črko ter pokliče XWORD

morse\_blink:

mov r0, #1000

bl DELAYTC

ldr r0, =NIZ

bl XWORD

b \_end

//Iz naslova v pomnilniku bere črko za črko in za vsako kliče GETMCODE ter XMCODE

XWORD:

push {r1, lr}

loop\_word:

ldrb r1, [r0]

cmp r1, #0

beq exit\_XWORD

push {r0}

ldrb r0, [r0]

bl GETMCODE

bl XMCODE

pop {r0}

add r0, r0, #1

b loop\_word

exit\_XWORD:

pop {r1, pc}

//Podano črko spremenit v naslov, kjer se nahaja njena morse koda

GETMCODE:

push {r1, r2, lr}

ldr r1, =A

ldrb r1, [r1]

sub r1, r0, r1 //Naložim črko A ter jo odštejem podani črki

mov r2, #6

mul r1, r1, r2 //V r1 naložim odmik odmik črke v abecedi

ldr r0, =MORSE

add r0, r0, r1 //naslovu MORSE tabele prištejem odmik, da pridem do podane črke

pop {r1, r2, pc}

//zažmiga eno črko

XMCODE:

push {r0, r1, r2, lr}

loop\_letter:

ldrb r1, [r0]

cmp r1, #0

beq exit\_XMCODE

push {r0}

ldrb r0, [r0]

bl XMCHAR

pop {r0}

add r0, r0, #1

b loop\_letter

exit\_XMCODE:

pop {r0, r1, r2, pc}

//zažmiga en znak

XMCHAR:

push {r0,r1, r2, lr}

ldr r1, =PIKA

ldrb r2, [r1]

cmp r0, '.'

it eq

moveq r0, #150

it ne

movne r0, #300

bl LED\_ON

bl DELAYTC

bl LED\_OFF

mov r0, #150 //pause for 150

bl DELAYTC

pop {r0, r1, r2, pc}

//Spremenil na zeleno diodo

INIT\_IO:

push {lr}

ldr r6, = RCC\_AHB4ENR // Load peripheral clock reg address to r6

ldr r5, [r6] // Read its content to r5

orr r5, #0x00000200 // Set bit 8 to enable GPIOJ clock

str r5, [r6] // Store result in peripheral clock register

//2. MODER (Mode Register): 01: General purpose output mode

// Make GPIOJ Pin2 as output pin (bits 4:5 in MODER register)

ldr r6, =GPIOJ\_BASE // Load GPIOD BASE address to r6

ldr r5, [r6,#GPIOx\_MODER] // Read GPIOD\_MODER content to r5

and r5, #0xFFFFFFCF // Clear bits 4-5 for P2

orr r5, #0x00000010 // Write 01 to bits 4-5 for P2

str r5, [r6] // Store result in GPIO MODER register

pop {pc}

LED\_ON:

push {r5, r6, lr}

// Set GPIOx Pins to 0 (through BSSR register)

ldr r6, =GPIOJ\_BASE // Load GPIOJ BASE address to r6

mov r5, #LEDs\_ON

str r5, [r6,#GPIOx\_BSRR] // Write to BSRR register

pop {r5, r6, pc}

LED\_OFF:

push {r5, r6, lr}

// Set GPIOx Pins to 1 (through BSSR register)

ldr r6, =GPIOJ\_BASE // Load GPIOJ BASE address to r6

mov r5, #LEDs\_OFF

str r5, [r6,#GPIOx\_BSRR] // Write to BSRR register

pop {r5, r6, pc}