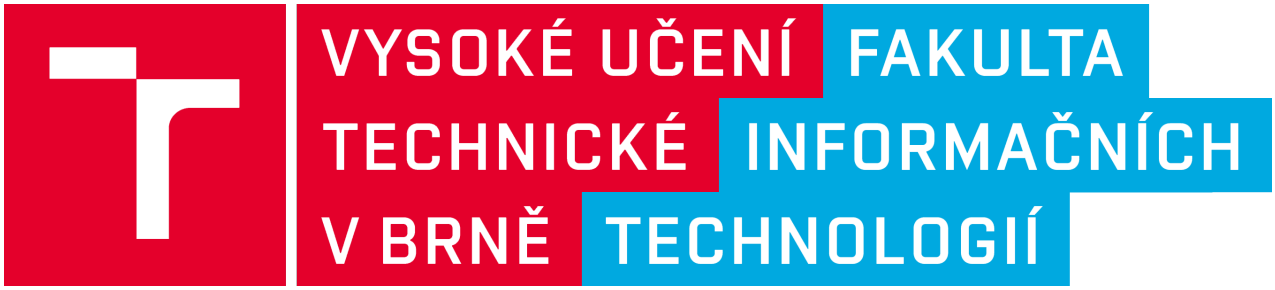


DOKUMENTÁCIA PROJEKTU

Varianta:Aplikace modulu Watchdog Timer (WDOG)

Platfotma: Arduino Uno

Mikročip: ATmega328p

Autor: Samuel Líška

Login: xliska20

13. decembra 2020

OBSAH

1. CIEĽ PROJEKTU

Vytvořte **projekt demonstrující možnosti modulu Watchdog Timer (WDOG)** dostupného na mikrokontroléru Kinetis K60 z desky platformy FITkit 3.

2. ZÁKLADNÉ INFORMÁCIE

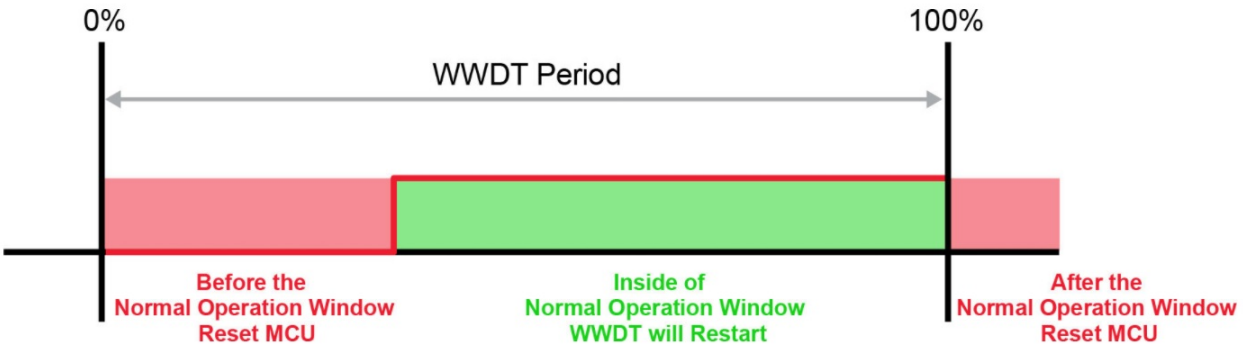
Z dôvodu pandemických okolností bol projekt realizovaný na platforme Arduino. Konkrétne sa jedná o model Arduino Uno s veľmi znamym mikročipom ATmega328p. Čip obsahuje Watchdog modul s nezávyslím oscilátorom. Projekt mal využívať 2 typy watchdogu, konkrétne periodický a okienkový.

a. Základné pojmy

Watchdog je modul, ktorý slúži ako prevencia “zaseknutia” alebo “zacyklenia” programu. Je to mechanizmus, ktorý mikroprocesor v prípade akéhokoľvek „zaseknutia“ programu vyvedie z tohto nežiadúceho stavu resetom prípadne prerušením.

i. **Periodický watchdog** - Watchdogu je zadané časové kvantum, počas ktorého je nutné inkrementovať čítač. Pokiaľ ostane čítač neinkrementovaný watchdog zasiahne resetom prípadne prerušením.

ii. **Okienkový watchdog** - Detekuje kratšie aj dlhšie časové cykly narozdiel od periodického. Pokiaľ sa časové kvantum nachádza v **otvorenom okne**, nieje nutné aby watchdog zasahoval, v opačnom prípade ak sa nachádza v **zatvorenom okne** je nutný zásah watchdogu.



3. NÁVRH

Dobrou analógiou fungovania watchdogu je tlačidlo bdelosti rušňovodiča. Funguje to presne rovnako. Ak je rušňovodič v poriadku, to znamená, že je bdelý a neprišlo mu zle, stláča v periodických intervaloch tlačidlo. Ak by dostal napríklad mikrospánok a tlačidlo v stanovenom časovom limite nestlačil, spustí sa núdzový mechanizmus. Najskôr sa riadiaci systém lokomotívy pokúsi nejakým signálom rušňovodiča prebudiť a ak sa to nepodarí, aktivuje sa brzdiaci mechanizmus, ktorý zastaví vlak.

a. Periodický watchdog

V periodických intervaloch je nutné stláčať tlačidlo aby sa vyresetoval watchdog. Ak sa tlačidlo nestlačí, program varuje rušňovodiča číslom na displeji.

b. Okienkový watchdog

Periodické riešenie funguje v danom príklade dobre, avšak dá sa oklamať držaním tlačidla ktoré by stále resetovalo watchdog. Preto je vhodné využiť **windowed watchdog** ktorý zaručí že pre úspešný reset je nutné aby sa reset vykonal v **otvorenom** stave čím by sa zabránilo príliš včasnému slačeniu tlačidla.

Po nevčasnej obsluhe sa zavolá prerušenie, ktoré je nutné obslúžiť aby vlak nezastavil

4. POPIS OBVODU A APLIKÁCIE

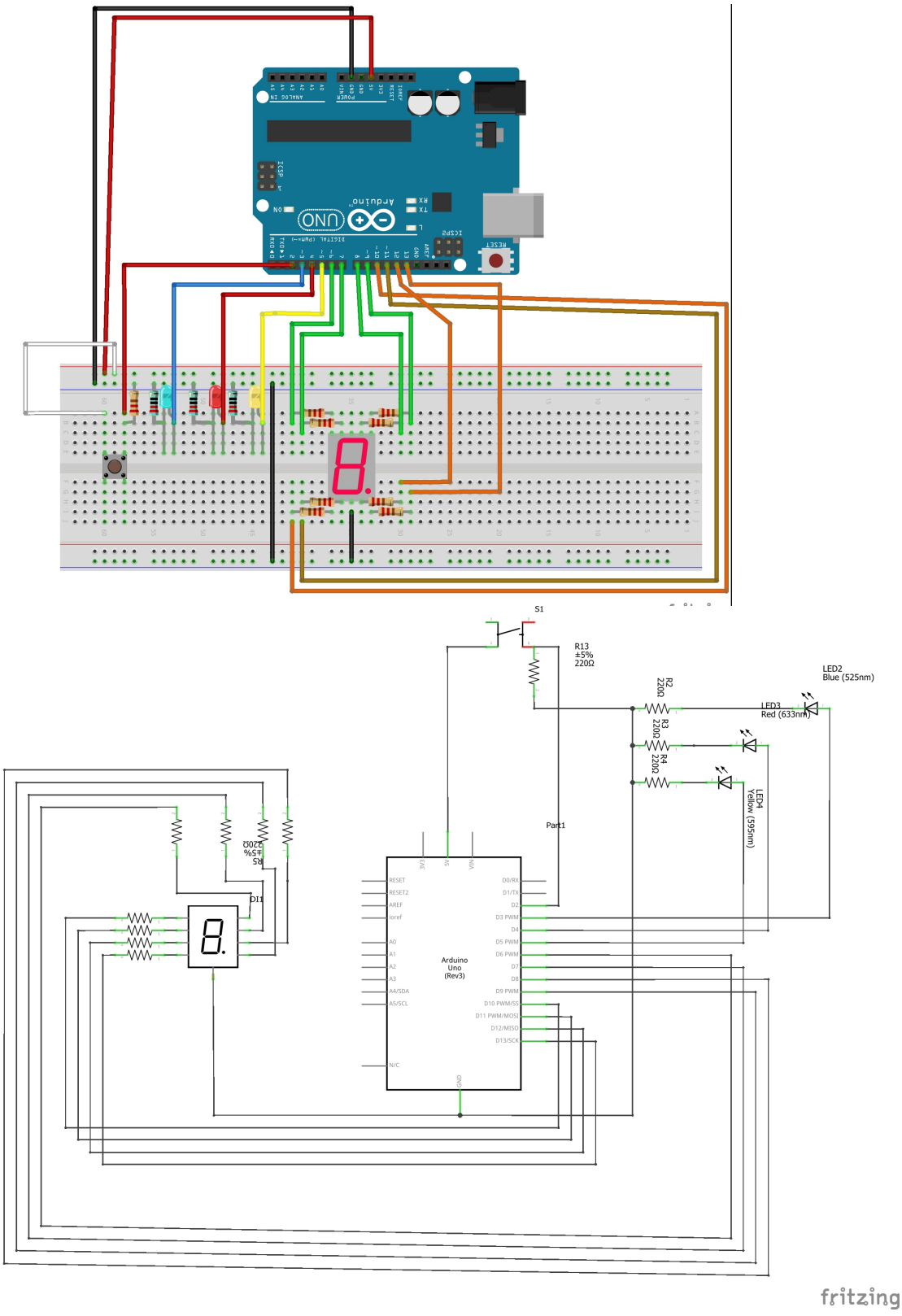
a. Stavy aplikácie

1. **Červená LED -** jazda vlaku(blikajúca)
2. **Modrá LED -** reset aplikácie(preblikne)
3. **Žltá LED -** prerušenie(svieti počas prerušenia)
4. **Displej -** čísla indikujú počet premeškaných resetov za sebou
5. **Všetky led a symbol ‘ - ’ na displeji** - vlak zastavuje

b. Funkčnosť aplikácie

Po zapojení a inicializácií sa spustí hlavná smyčka programu na ktorú upozorňuje blikajúca červena LED(jazda vlaku). Na jednocifernom displeji sa zobrazuje počet “nestihnutých” intervalov. Pri kliknutí tlačidla sa spustí wd_reset, čím sa počet vynuluje. Ak ale počet prekročí 3, watchdog sa nastaví do **interrupt režimu(svieti žltá led)**. Následne sa na displeji zobrazí náhodné číslo od 0 do 9(posledná možnosť vodiča). Vodič má 7 sekúnd na zadanie daného čísla do terminálu aby zabránil **zastaveniu vlaku(svietia všetky 3 led a na displeji je znázornený znak mínus ‘ - ‘)**.

c. Schéma zapojenia



d. Arduino a Watchdog

Chovanie watchdog-u sa v Arduine nastavuje pomocou WDTCSR registra. A dĺžka intervalu pomocou WDP registra(prvé 4 bity).

WDTCSR – Watchdog Timer Control Register

Bit	7	6	5	4	3	2	1	0	
(0x60)	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0	WDTCSR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	X	0	0	0	

WDTON[1]	WDE	WDIE	Mode	Action on Time-out
1	0	0	Stopped	None
1	0	1	Interrupt Mode	Interrupt
1	1	0	System Reset Mode	Reset
1	1	1	Interrupt and System Reset Mode	Interrupt, then go to System Reset Mode
0	x	x	System Reset Mode	Reset

WDP 3	WDP 2	WDP 1	WDP 0	Time-out (ms)
0	0	0	0	16
0	0	0	1	32
0	0	1	0	64
0	0	1	1	125
0	1	0	0	250
0	1	0	1	500
0	1	1	0	1000
0	1	1	1	2000
1	0	0	0	4000
1	0	0	1	8000

5. IMPLEMENTÁCIA

Použité knižnice

<avr/wdt.h> - knižnica pre watchdog timer(väčšinu času som ale používal priamo registre)

<EEPROM.h> - knižnica pre nevolatívnu pamäť do ktorej sa ukladaly stavy po resetoch

a. void setup()

Funkcia **setup()** je prvá funkcia ktorá sa vykoná pri spustení alebo po resete arduina. Vo funkcii je okrem potrebných inicializácií obvodu využita **EEPROM.h** knižnica, ktorá umožňuje ukladať hodnoty do nevolatívnej pamäte, takže po reštarte aplikácie dôležité štatistické data zostanú zachované a dá sa snimi opäť pracovať. Konkrétne sa jedná o premennú **WD_count** ktorá drží počet reštartov od posledného stlačenia tlačidla. Ak tento počet prekročil 3, funkcia watchdog sa nastaví na režim **interrupt**

b. void loop()

Hlavné telo aplikácie, ktoré sa vykonáva dookola. V mojom prípade iba simuluje jazdu vlakom(vypínanie a zapínanie červenej LED)

c. void watchdogSetup()

Stará sa o správne nastavenie Watchdog modulu. Berie 1 argument v ktorom je mód v akom má watchdog následujúci reset bežať. Nastavuje konkrétne **WDTCSR** register. Na začiatku funkcie sa vypnú prerušenia a na konci sa opäť zapnú aby nedošlo k chybe.

WDTCSR register:

WDIF = Interrupt flag(nastavuje systém)

WDIE = Zapnúť prerušenie

WDP3 = 2000ms Time-out

WDCE = konfiguračný mód

WDE = Zapnúť reset

WDP2 = 2000ms Time-out

Bit	Name
7	WDIF
6	WDIE
5	WDP3
4	WDCE
3	WDE
2	WDP2
1	WDP1
0	WDP0

WDP1 = 2000ms Time-out

WDP0 = 2000ms Time-out

d. ISR()

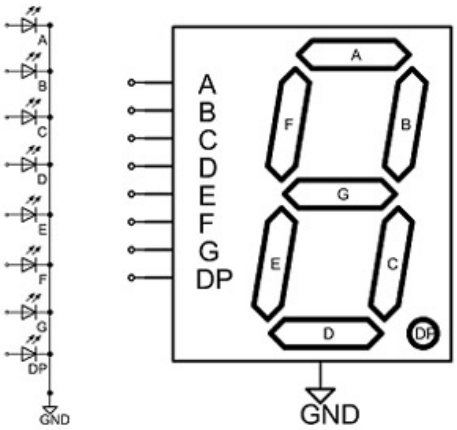
Interrupt service routine. Funkcia ktorá obsluhuje prerušenia. Pretože sa odporučuje obsluhu prerušenia vyriešiť čo najrýchlejšie, táto funkcia iba nastavi **interrupt** flag, aby sa pri následnom reštarte mohlo interakovať s rušňovodičom.

e. void interruptVerify()

Funkcia vytvára kontrolnú otázku. Zobrází číslo na displej ktoré treba do 7 sekúnd napísať do terminálu. Po uplynutí času alebo zlej odpovede sa spustí zastavovanie vlaku. V prípade správnej odpovede sa nastaví opätovný reštart vo watchdogu a do nevolatívnej pamäte sa uloží vynulovaný counter.

f. displayNumber()

Volanie funkcií pre výpis konkrétnych čísel na displej.



6. ZÁVER

a. Zhodnotenie návrhu a implementácie

Z pandemických dôvodov bol projekt realizovaný na platforme Arduino Uno, na ktorej bohužial' nieje možné využiť **okienkový** watchdog timer. Z tohodo dôvodu táto časť projektu nieje implementovaná, a nieje ošetrené “držanie” tlačidla pre resetovanie watchdogu(takže rušňovodič môže držať dlačidlo bez toho aby dával pozor).

Na druhú stranu bolo nutné celý obvod ručne zapájať, keďže implicitne je toho na doske len veľmi málo(1 červená LED na pin 13). Preto som si musel zapojenie najskôr naštudovať, a postupne zapájať a programovať.

b. Video prezentácia

https://nextcloud.fit.vutbr.cz/s/oL5PFoLjR7iaAZT

c. Zdroje

- Prednášky IMP
- https://www.arduino.cc/reference/en/libraries/watchdog/
- https://dlnmh9ip6v2uc.cloudfront.net/learn/materials/8/Arduino_Cheat_Sheet.pdf
- https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf