Mikropočítačové Systémy

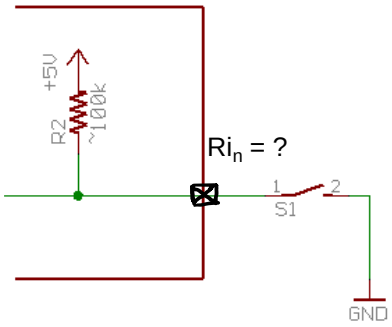
MIPS

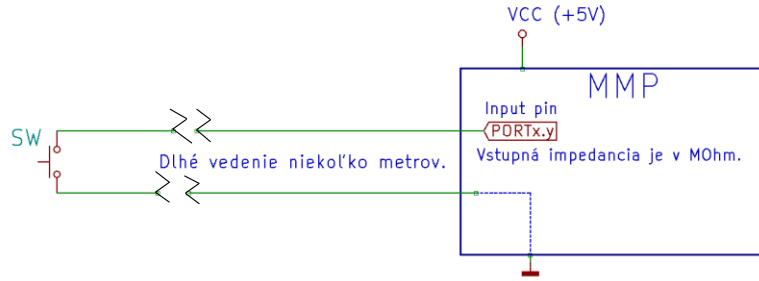
Prednáška 4.
Binarne vstupy.

Treba niečo „počítať“, ak výstupom kontaktu je len Zap/Vyp?
Derivácia (diferencia) je tu použitá implicitne – explicitne.

I/O Príklad: jednoduchý kontakt

Netreba pripájať žiadne externé pullup rezistory





Je to pravda? Platí to vždy a všade?

2

Ešte na prvej prednáške sme si ukázali, aké je to jednoduché, keď chceme realizovať binárny vstup. Nakonfigurujeme daný pin portu ako vstupný, a je to.

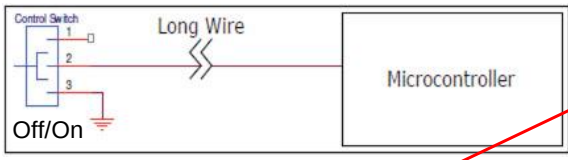
Prax niekedy od nás požaduje tzv. vzdialený zdroj signálu. Za takéto môžeme považovať aj pripojenie kontaktu viac ako 5 m dlhým vedením.

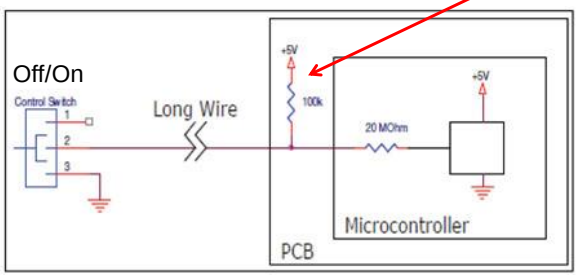
Tento príklad predpokladá napájanie procesora VCC = 5V. Vstupnú impedanciu pinu niekoľko MOhm. Tento údaj nenájdeme v KL, ale nájdeme tam Input Leakage Current I/O Pin (Low/High) max. value 1uA.

Keďže jeden krát zažiť je viac ako niekoľko krát čítať pokúsim sa zopakovať príklad z internetu. Musel som ho trocha modifikovať. Dôvod: Doma nemám až tak kvalitné vybavenie. Našiel som len 5m tzv. telefónnej dvojlinky a osciloskop s parametrami: 10 MHz a vstupnou impedanciou sondy 1MOhm.

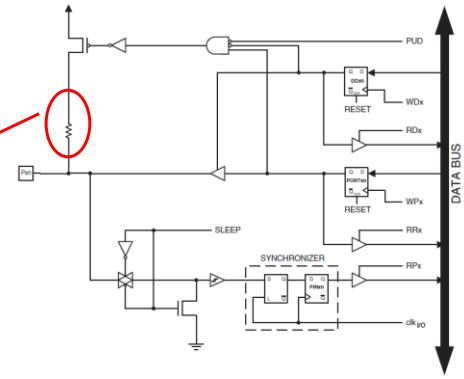
Ochrana I/O vstupov mikropočítača

Predpokladajme pripojenie I/O pinu k tlačítku niekoľko metrov dlhým vodičom.





R_{PU} =20 – 50kΩ

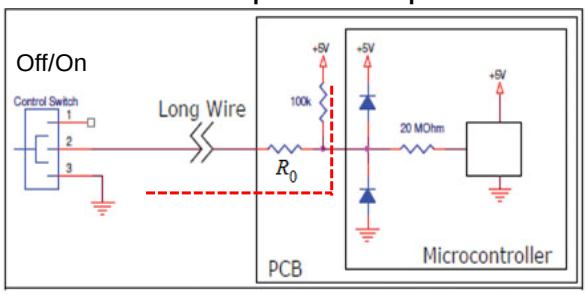


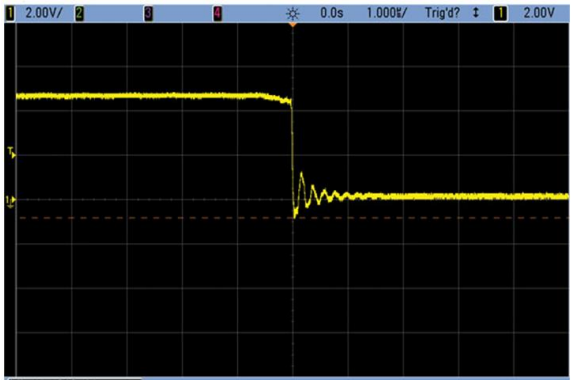
Takýto priebeh by mohol zničiť mikropočítač

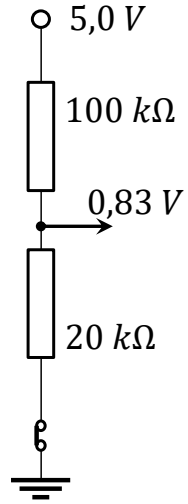
3

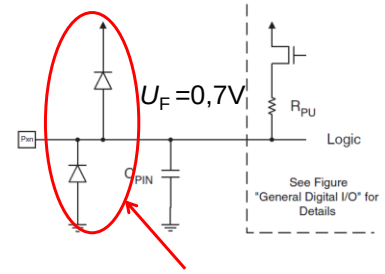
AVR majú zabudovaný pullup 20-50 kOhm. Pri niektorých typoch MMP je to aj 100 kOhm. Cez takýto rezistor napájame linku, ktorá sa javí ako RLC záťaž. V prípade aktivovania tlačítka vznikne prechodný proces, ktorý môže zničiť vstupné obvody MMP.

Ochrana I/O vstupov mikropočítača: Obmedzenie prúdu.









V ideálnom prípade takáto ochrana postačuje. Avšak ak je napätie veľké alebo ak trvá dlho, môžu sa diódy zničiť, dokonca aj vnútorné obvody mikroprocesora ...
Dokonca aj keď sa diódy nezničia, veľké ESD impulzy môžu vyvolať ***veľké prúdy v napájaní*** mikroprocesora. Toto je ďalší možný zdroj porúch.
To znamená, treba obmedziť prúd. Pridáme do vstupu rezistor R_0.
R_0 je treba navrhnúť tak, aby na ňom nevznikol veľký úbytok napätia. Rezistory 100kΩ a R_0 tvoria napäťový delič.

4

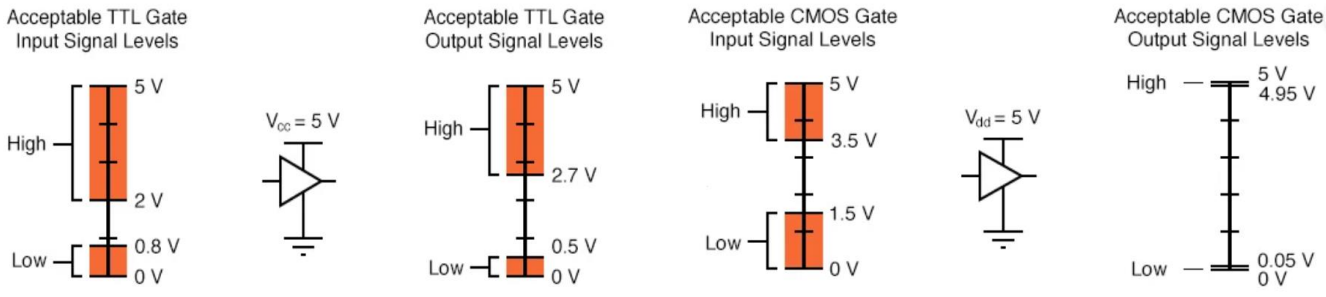
Rezistory 100 kOhm a 20 kOhm tvoria odporový delič.

Ak by sme napr. zvolili R_0 = 20 kOhm (2*10 kOhm), zopnutému kontaktu by odpovedalo napätie 0,83V, čo je oveľa viac ako TTL log. úroveň LOW.

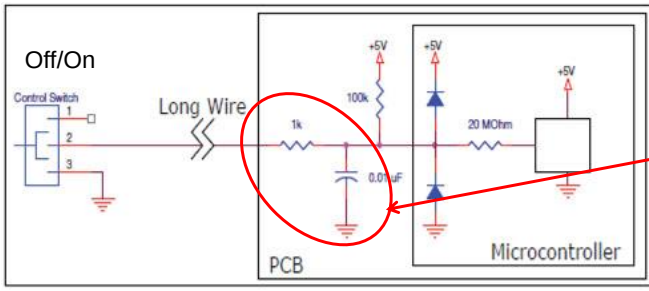
Ako je vidieť z obr. dole vľavo, napätie na vstupe neklesne pod hodnotu cca –0,7V.

TTL

CMOS


Acceptable TTL Gate
Input Signal Levels
5 V
High
2 V
0.8 V
Low
0 V
V_{cc} = 5 V
Acceptable TTL Gate
Output Signal Levels
5 V
High
2.7 V
0.5 V
Low
0 V
Acceptable CMOS Gate
Input Signal Levels
5 V
High
3.5 V
1.5 V
Low
0 V
V_{dd} = 5 V
Acceptable CMOS Gate
Output Signal Levels
High
5 V
4.95 V
Low
0.05 V
0 V

Ochrana I/O vstupov mikropočítača
Filtrácia



Kvalitnejší spôsob ochrany získame, ak pridáme kondenzátor.

Vytvoríme RC filter.

Obvod obmedzujúci prúd sme upravili na dolnopriepustný filter.

Pri výbere rezistrora a kondenzátora musíme zobrať do úvahy frekvenčné vlastnosti obvodu. Filter nesmie odfiltrovať užitočné signály.

6

Ináč povedané, mali by sme uvažovať prenosové vlastnosti vstupného kanála. Len málokedy sa v KL objavuje pojem „pásmo priepustnosti“. Namiesto neho prax používa pojmy Rise Time, resp. Fall Time. T.j. ako rýchle signál prejde z jednej úrovne do druhej. A samozrejme nami navrhnuté zariadenie by malo vedieť tieto signály sledovať. Nemalo by ich odfiltrovať.

BandWidth – Rise/Fall Time

LM358
LOW POWER DUAL OPERATIONAL AMPLIFIERS

Features

- Wide bandwidth (unity gain): 1MHz (temperature compensated)

DRV5055 Ratiometric Linear Hall Effect Sensor

1 Features

- Fast 20-kHz Sensing Bandwidth

AD 10 bits ≡ 5mm

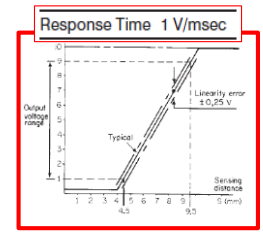
Volný pád: $h(t) = h_0 - \frac{1}{2}gt^2 \Longrightarrow$

$$t_{1\,bit} = \sqrt{\frac{0.005}{2^{10}}\frac{2}{g}} \approx 0.99\,ms$$

$$t_{10\,bit} = \sqrt{\frac{0.005}{2^{0}}\frac{2}{g}} \approx 32\,ms$$

Proximity Sensors

30 mm (1.18 in.)



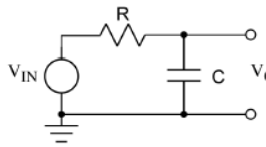
7

Vlastnosti OZ-ov, snímačov polohy, sú v KL popisované „trocha“ ináč ako sa to zvykne prezentovať na teoretických predmetoch typu:

Teória riadenia.

Ochrana I/O vstupov mikropočítača

Súvis medzi „BandWidth – RiseTime“ = „Šírka pásma – Doba nábehu “



$Z_R = R$

$Z_C = 1/(j\omega C)$

Prenos napäťového deliča je:

$$H(j\omega) = \frac{V_{OUT}}{V_{IN}} = \frac{Z_C}{Z_C + Z_R} = \frac{1}{1 + j\omega RC}$$

Dosaďme: $\omega = 2\pi f$ a $T = R.C$

Potom: $H(f) = \frac{1}{1 + j2\pi f RC}$

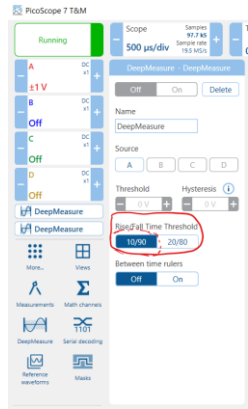
Frekvencia f na ktorej modul $H(f)$, t.j. $|H(f)| = 1/\sqrt{2}$ poklesne o -3*dB* sa nazýva pásmo priepustnosti.

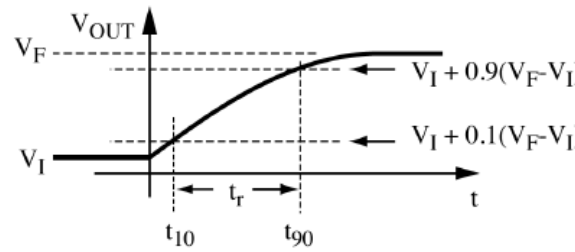
„RiseTime“ je čas za ktorý prejde V_{OUT} z 0.1(V_f - V_I) na 0.9 (V_f - V_I).

Výsledkom je : $t_r = 2.2T$

Po dosadení: $T = RC \overset{!}{=} \frac{1}{2\pi f}$

$$t_r = 0.35\frac{1}{f_{3dB}}$$

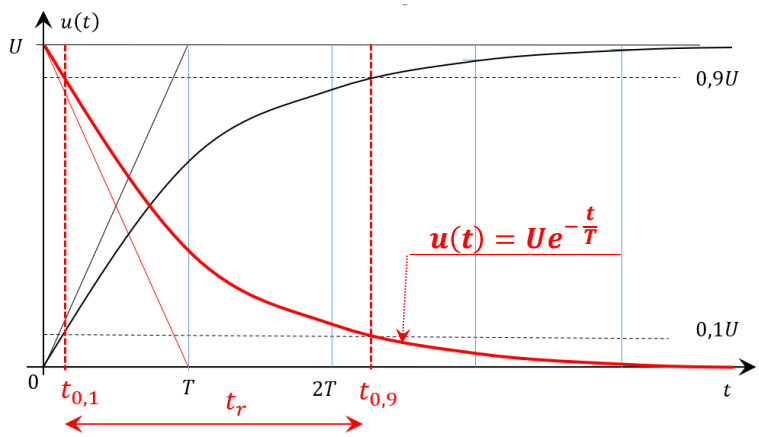




8

Predpokladáme ideálny napäťový zdroj. Má nulový vnútorný odpor. Napätie na kondenzátore bude nabiehať po exponenciále.

RiseTime“ = „Šírka pásma – Doba nábehu “: Riešenie



„RiseTime“ je čas za ktorý prejde

V_{OUT} z 0.1(V_f - V_l) (U) na 0.9 (V_f - V_l) (U).

$$0,9U = Ue^{-\frac{t_{0,1}}{T}} \qquad 0,1U = Ue^{-\frac{t_{0,9}}{T}}$$

$$\frac{0,9}{0,1} = \frac{e^{-\frac{t_{0,1}}{T}}}{e^{-\frac{t_{0,9}}{T}}} \qquad \ln 9 = (t_{0,9} - t_{0,1})\frac{1}{T} = t_r\frac{1}{T}$$

$$t_r \doteq 2,2T \qquad T = R.C \qquad \omega = 2\pi f$$

$$t_r = 0.35\frac{1}{f_{3dB}}$$

9

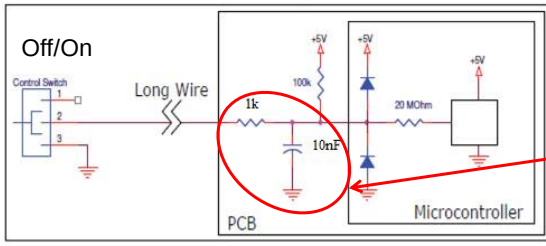
Zhrnutie: Ak máme napr. uvedené, že tzv. Rise Time zdroja signálu je 10 us, potom by sme mali zvoliť pásmo priepustnosti napr. 5*väčšie.

T.j. f_{3dB} = 0,35/(2us) = 175 kHz. A potom R*C = 1/(2*pi*175000) =0,91 us.

Ak zvolíme R = 1 kOhm, potom C (približne) = 1 nF.

Ochrana I/O vstupov mikropočítača

Filtrácia **, pokračovanie**



Predpokladajme $t_r = 20\ \mu s$

To odpovedá $f_{3dB} = 0{,}35\dfrac{1}{20\mu s} = 17{,}5kHz$

Zvoľme $R = 1k\Omega$

potom $C = \dfrac{t_r}{2{,}2}\dfrac{1}{R} = \dfrac{20\ \mu s}{2{,}2}\dfrac{1}{1k\Omega} = 9{,}1\ nF$

a zaokrúhlime $C \cong 10\ nF$

Pridanou hodnotou

RC filtra

je skutočnosť, že RC filter potlačí náhodné, neželané krátke vstupy, ktoré by mohli spôsobiť chybnú prácu mikropočítača.

Poznamenajme, že pre veľké ESD a dlhé vedenia ani toto nemusí postačovať.

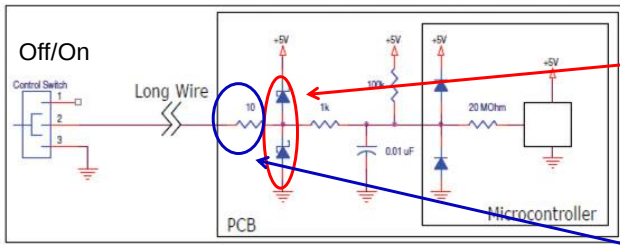
Ochrániť vnútorné diódy mikropočítača môžeme pomocou externých **Schottky-ho diód**.

10

Príklad: Je dané t_r a vypočítajme R a C.

Ochrana I/O vstupov mikropočítača

Filtrácia, Schottky-ho diódy



Schottky-ho diódy

sa používajú preto, že vedú prúd skorej ako vnútorné ochranné diódy. (úbytok Schottky-ho diód v priepustnom smere je cca 0,2 V. Úbytok vnútorných diodách je cca 0,7 V).

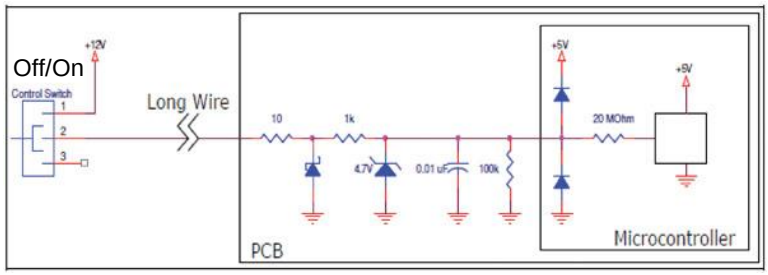
Malý sériový rezistor

je použitý na ochranu Schottky-ho diód pred veľkým prúdom. Ak sú diódy v stave zopnutom len krátko, postačuje rezistor veľkosti 10 Ω.

.

Ochrana I/O vstupov mikropočítača

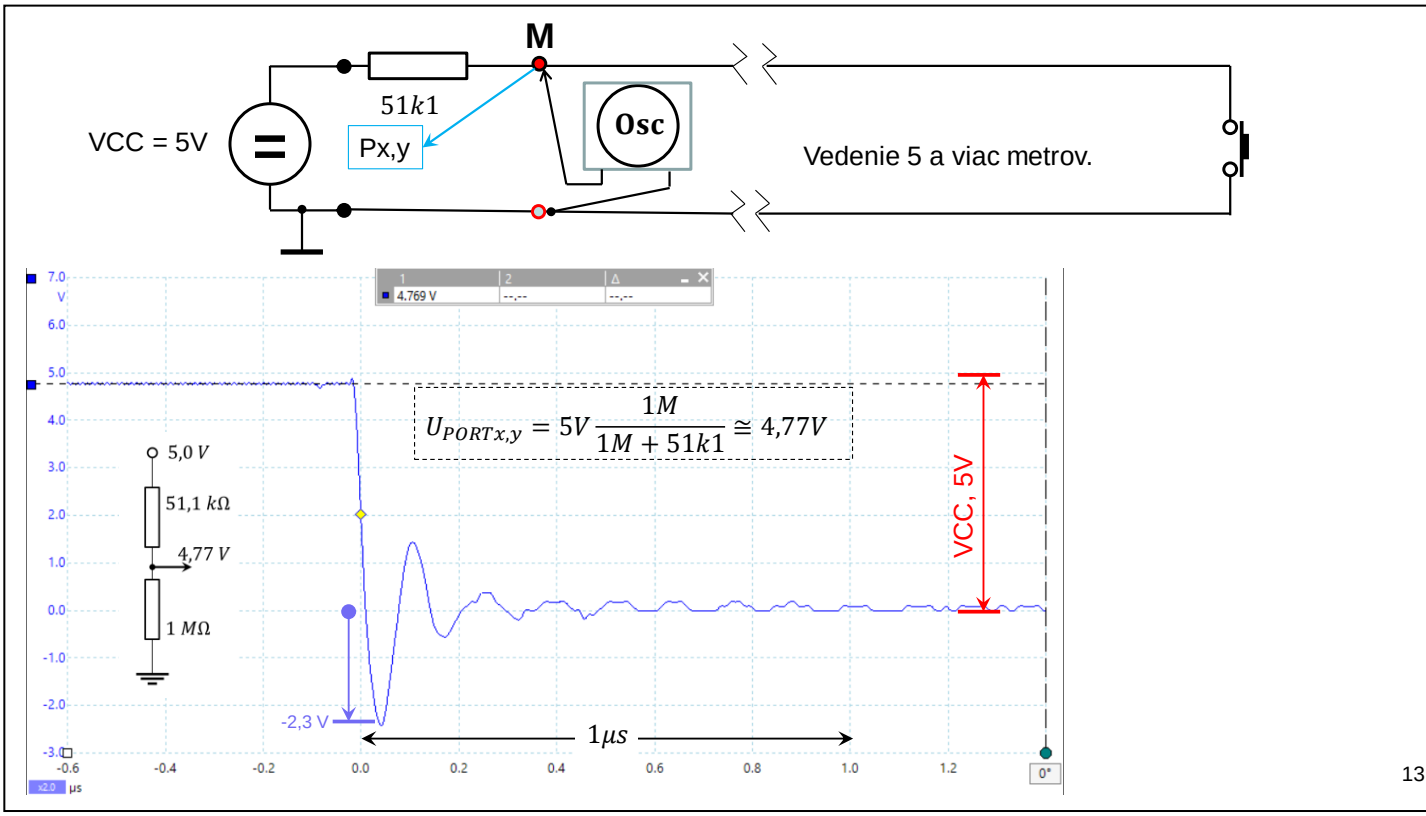
Iné možné riešenia



I/O pin je cez tlačítko pripojený na napätie 12V. V tomto prípade kontakt nespína I/O pin **k zemi**, ale k **12V-om**.
Schottky-ho dióda obmedzuje záporné napäťové špičky.

Zenerová dióda, pred ktorou je zapojený rezistor obmedzujúci prúd. Výstupom zenerovej diódy je napätie odpovedajúce log. 1.

Poznamenajme, že rezistor obmedzujúci prúd treba voliť dostatočne malý (cca 1 mA) na jednej strane, a na strane druhej zas tak veľký, aby sa zenerová dióda dostala do vodivej oblasti.



13

Príklad sme zopakovali, a trocha modifikovali. Použili sme tvrdý zdroj 5V. Namiesto tlačítka sme len vzdialené konce dvojlinky skratovali. Do plus napájania sme pripojili rezistor 51k1.

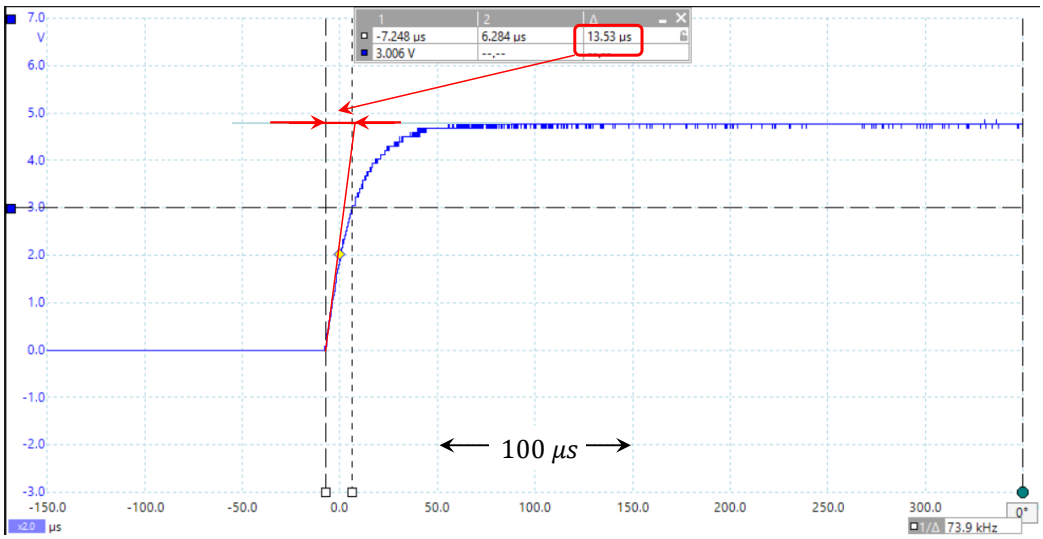
Vnútorný odpor sondy osciloskopu a 51k1 rezistor vytvorili odporový delič. V bode M sme odmerali pokles napätia na hodnotu 4,77V. T.j. zhoda teórie a praxe. Pri skrate vznikol prechodný proces odpovedajúci sústave 2. rádu kmitavej tlmenej.

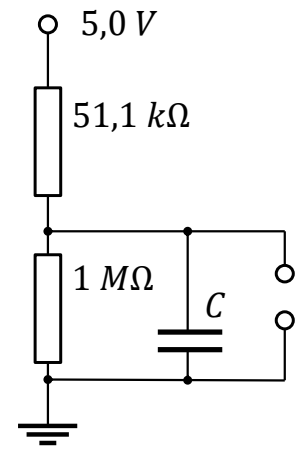
Časová konštanta: $\tau = RC = 13{,}5\mu s = 51k1 * C \implies C = 264\ pF$

Parametre sondy:
Bandwidth: 60MHz;
Rise Time: 5,8ns;

$$t_r = 0.35\frac{1}{f_{3dB}}$$

$264\ pF - 100pF\ (sonda\ tlmenie\ 1x) = 164\ pF$





14

Po „uvolnení tlačitka“ (rozopnutí kontaktu) sa linka opäť nabije. Ak zanedbáme vybíjanie „kondenzátora“ paralelne zapojenou impedanciou osciloskopu ku kondenzátoru, môžeme z exponenciálneho priebehu nábehu napätia určiť náhradnú kapacitu linky (Samozrejme, že súčasťou je aj vstupná kapacita sondy osciloskopu (cca 100 pF)).

Trvanie prechodného procesu (nábeh/dobeh) netrvá rovnako dlho.

Na meranie som použil sondu k osciloskopu: Model P7060
Bandwidth: 60MHz; Rise Time: 5.8ns; Attenuation Ratio (AR): 1x & 10x;

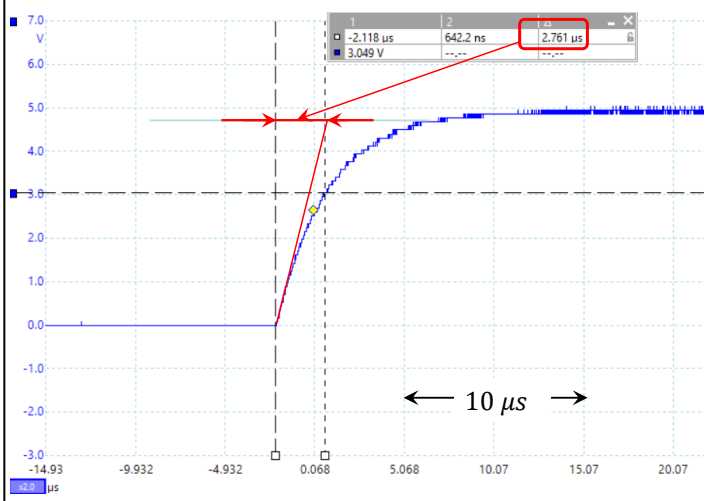
Input Resistance: AR = 1x,1MΩ±2%; AR = 10x, 10MΩ±2%
Input Capacitance: AR = 1x, 85pF~135pF ; AR = 10x,14pF~18pF

Máme dve možnosti, buď použijeme „lepší“ osciloskop, alebo zmeníme pullup rezistor.

Časová konštanta:

$\tau = RC = 2{,}76\mu s = 10k * C \quad \Longrightarrow C = 276\ pF$

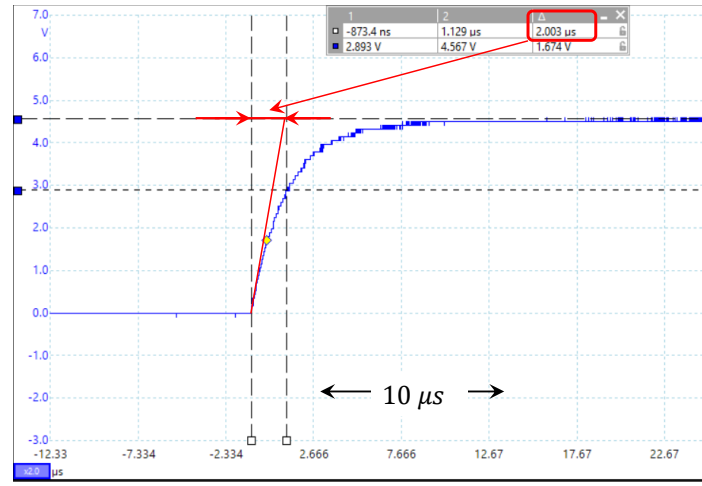
$276\ pF - (sonda\ tlmenie\ 1x)\ 100pF = 176pF$



Tlmenie 1x

$\tau = RC = 2{,}0\mu s = 10k * C \quad \Longrightarrow C = 200\ pF$

$200\ pF - (sonda\ tlmenie\ 10x)\ 16pF = 184\ pF$



Tlmenie 10x

15

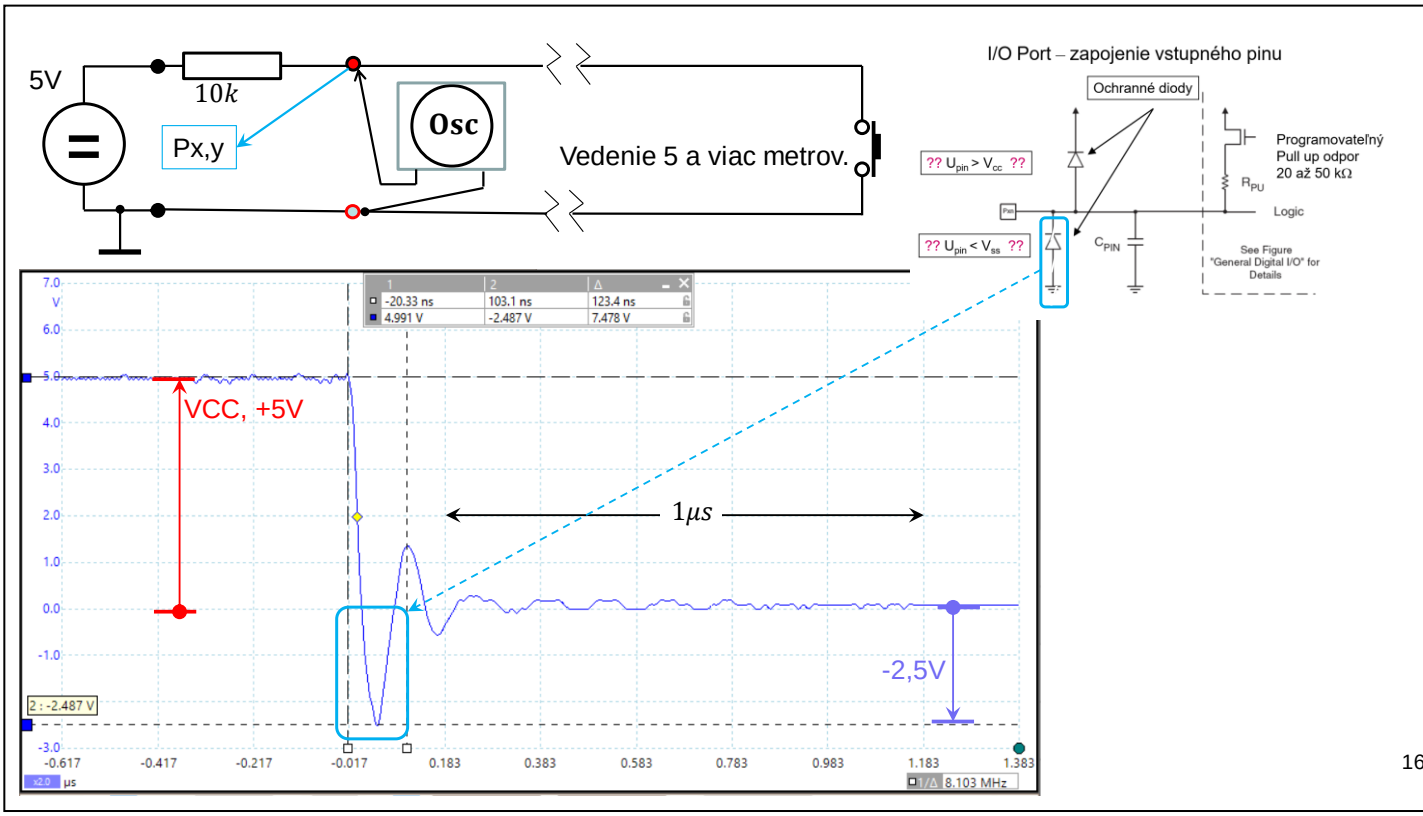
Použijeme informácie z poznámok k predchádzajúcej fólii.

Zmeníme hodnotu rezistora R = 10 kOhm. Zistime ako vplýva zmena tlmenia sondy (kapacita sondy) na prechodné procesy.

Obrázok naľavo: stredná hodnota kapacity sondy je 100 pF.

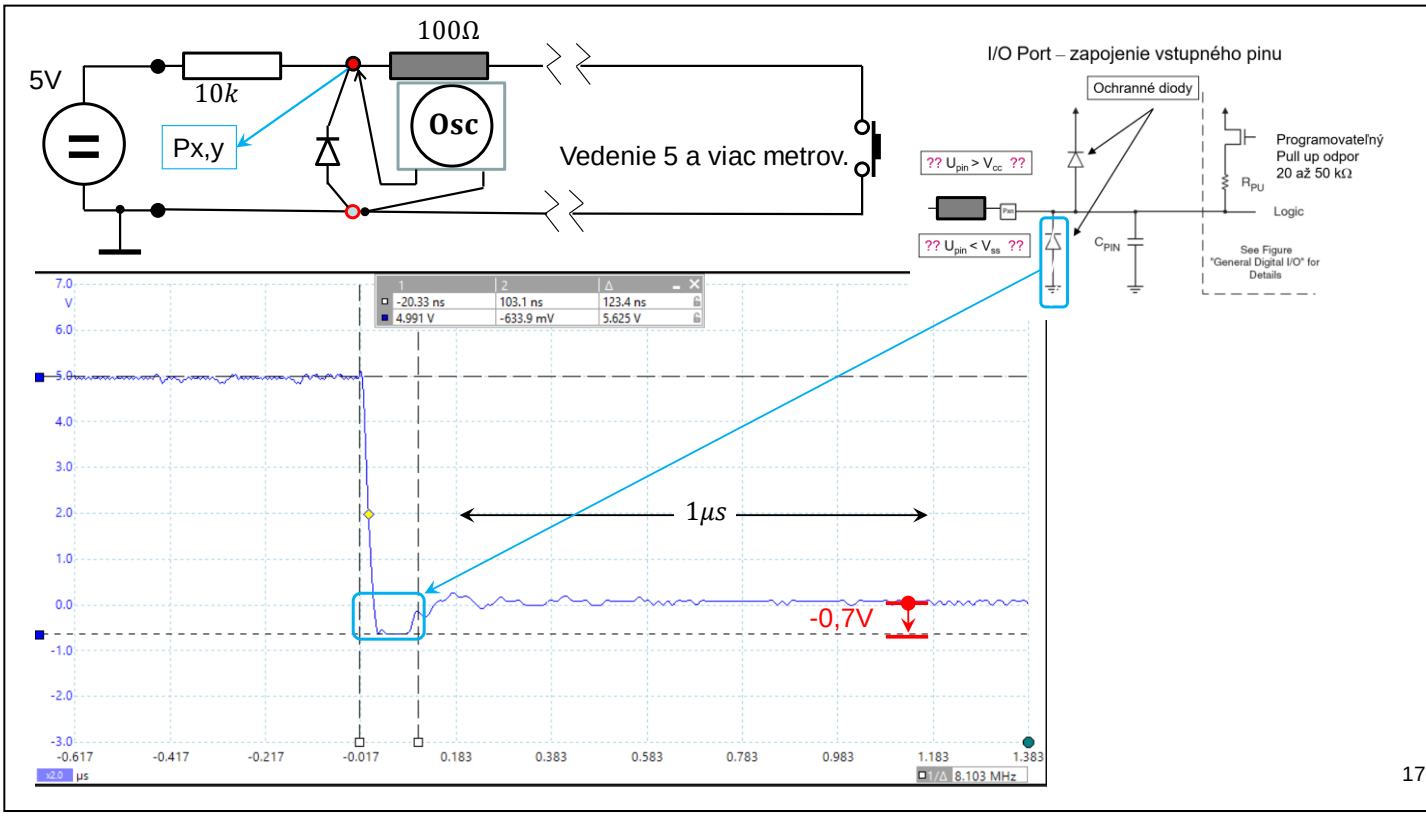
Obrázok napravo: stredná hodnota kapacity sondy je 16 pF.

Máme tri výsledky kapacity vedenia: 164pF, 176pF a 184pF. Meracím prístrojom sme odmerali 180,3 pF. Opäť dobrá zhoda teórie a praxe.



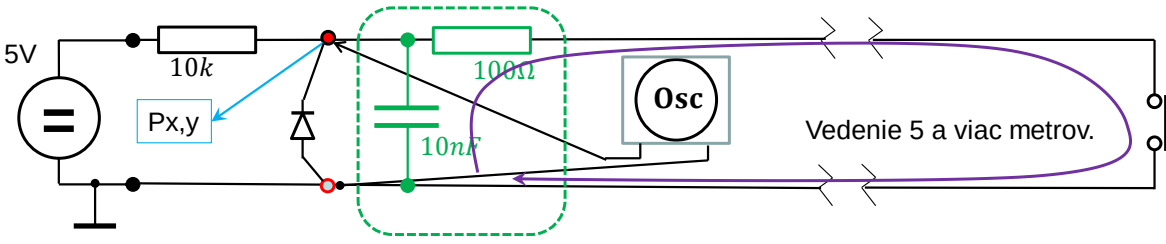
Vstupné napätie –2,5V môže zničiť vstupné obvody MMP.

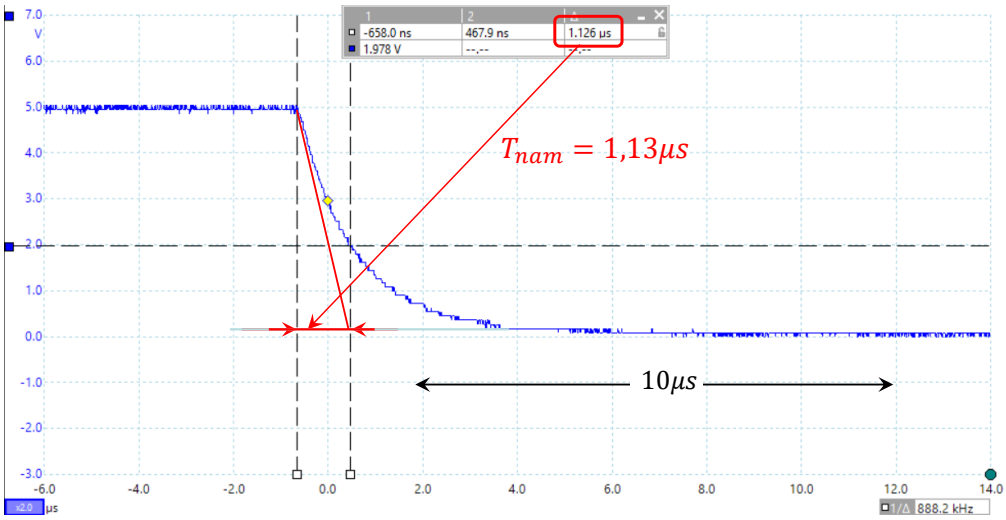
Pomôžeme ochrannou diódou.



Vstupné napätie –2,5V je „orezané“ ochrannou diódou.

Z KL zistíme, že prúd diódou treba obmedziť. Treba zapojiť do vstupu rezistor napr. 100 Ohm.





Časová konštanta filtra:

$T = RC =$
$=100\ \Omega * 10nF = 1{,}0\ \mu s$

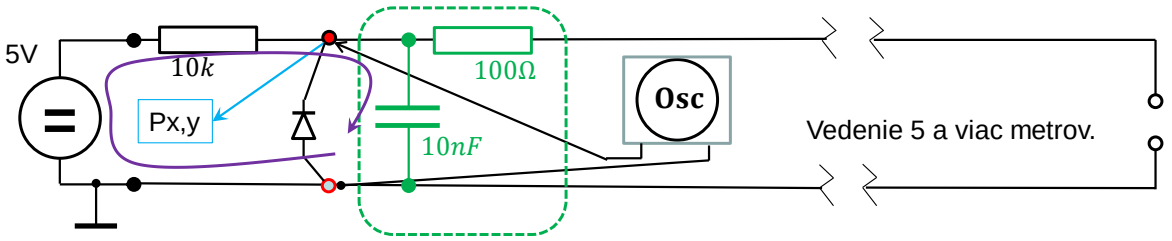
$t_r = 2{,}2T = 2{,}2\mu s$

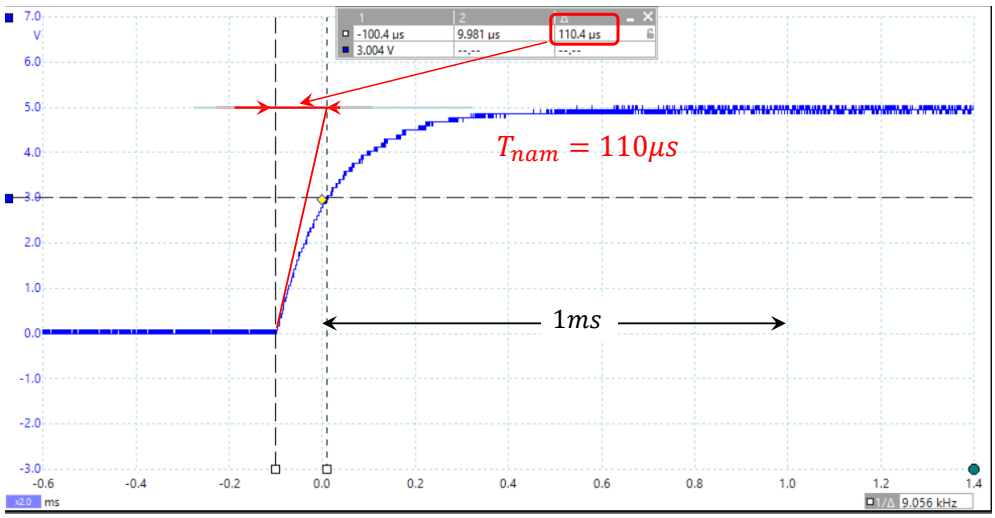
$f_{3dB} = \frac{0{,}35}{t_r} = 160kHz$

18

Nepekné priebehy prechodných procesov môžeme vylepšiť RC vstupným filtrom.

Doma nemám neobmedzený výber rezistorov a kapacitorov. Preto som nepostupoval tak, že je dané: Rise/Fall Time, resp. pásmo priepustnosti, ale vybral som rezistor 100 Ohm a kapacitor 10 nF a overil som namerané hodnoty.





Časová konštanta nabíjania:

$T = RC =$
$=10k\Omega * 10nF = 0{,}1\ ms$

$t_r = 2{,}2T = 0{,}22ms$

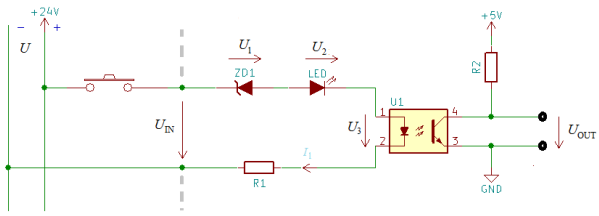
$f_{3dB} = \frac{0{,}35}{t_r} = 1{,}59\ kHz$

19

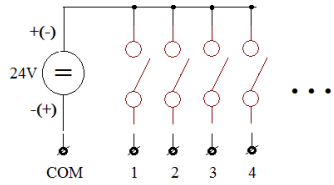
Nepekné priebehy prechodných procesov môžeme vylepšiť RC vstupným filtrom.

Doma nemám neobmedzený výber rezistorov a kapacitorov. Preto som nepostupoval tak, že je dané: Rise/Fall Time, resp. pásmo priepustnosti, ale vybral som rezistor 100 Ohm a kapacitor 10 nF a overil som namerané hodnoty.

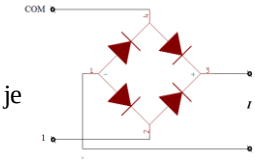
Poznamenajme nábeh/dobeh netrvá rovnako dlho.

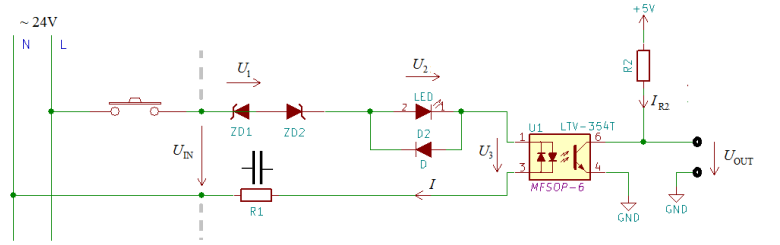


Rezistor R1 zabezpečuje v obvode prúd cca 10 mA.
Zenerová dióda ZD1 nastavuje hranicu medzi kontakt **ON/OFF** .

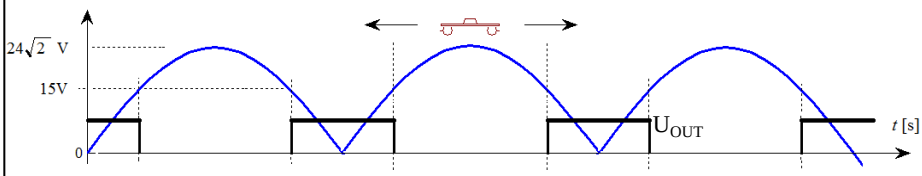


Riešením problému je





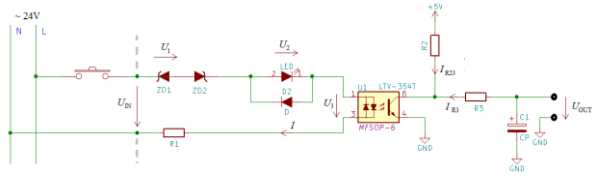
Aj napriek tomu, že je kontakt zopnutý,
Výstupný signál U_{out} je nejednoznačný.

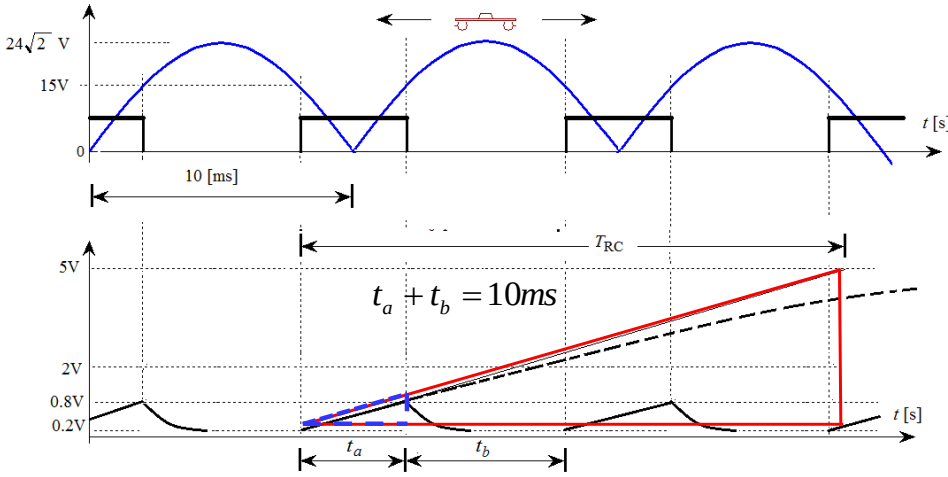


20

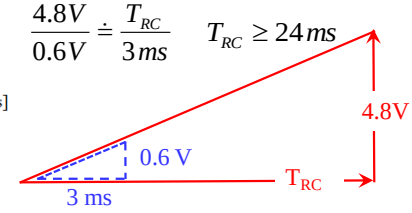
Ďalšou možnosťou ako pripojiť vzdialený kontakt je použiť optočlen. Galvanicky oddelíme obvody a naviac môžeme použiť aj iné napájanie kontaktu, dokonca aj striedavé napätie.

Príklad:
Filtrácia na výstupnej strane.
Kontakt: trvale zopnutý.





$$\frac{4.8V}{0.6V} \doteq \frac{T_{RC}}{3ms} \qquad T_{RC} \geq 24ms$$



$T_{RC} = (R_2 + R_3).C \Rightarrow$

Zvolíme $C = 1\mu F$ potom
$R_2 + R_3 = 24k\Omega$

t_a vypočítame: $24\sqrt{2} \doteq 34V$ zo vzťahu $\sin(2\pi f.\frac{t_a}{2}) = \frac{15}{34} \Rightarrow t_a \doteq 3ms$

Za čas t_a nesmie U_{OUT} opustiť „log. 0“ !!!!!!

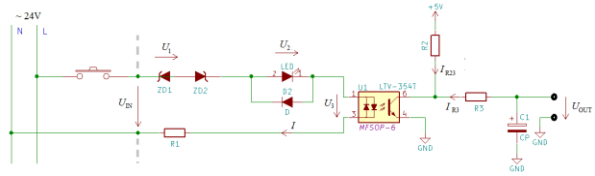
21

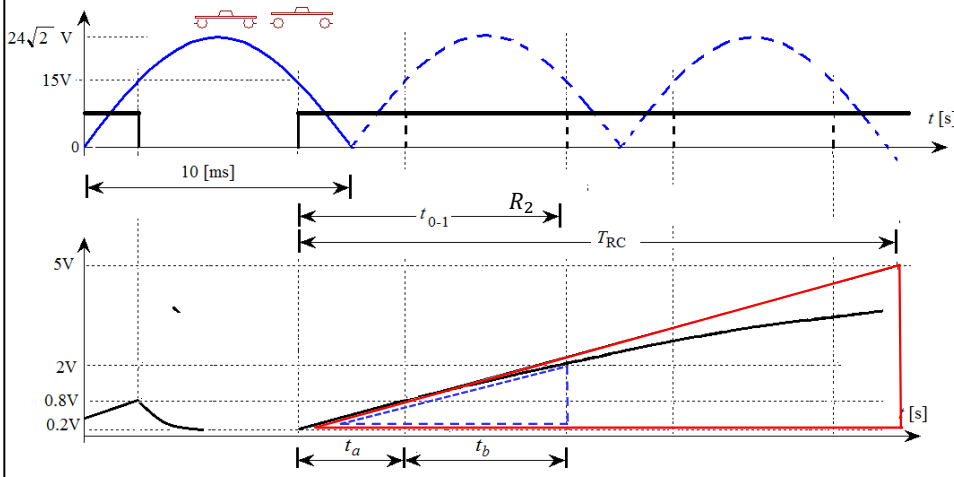
Naším cieľom, je navrhnúť filter na výstupe optočlena tak, aby reakčný čas na Zap/Vyp kontaktu bol čas jednej polperiody, t.j. 10ms.

Pri tomto riešení je to ako s 1. a 2. SV, resp. CISC a RISC. Prvé počítače nemali „nálepku“ CISC. Všetky boli tejto architektúry. Podobne to bolo v minulosti s týmto príkladom. Nemuseli sme zdôrazňovať, že v príklade sú použité úrovne TTL signálu.

Príklad:
Kontakt rozpojíme.

Za aký čas prejde U_{OUT} zo stavu „log. 0“ do stavu „log. 1“ ?





t_{0-1} vypočítame zo vzťahu: $\frac{4.8V}{1.8V} = \frac{T_{RC}}{t_{0-1}} \Rightarrow t_{0-1} = cca\ 9\,ms$

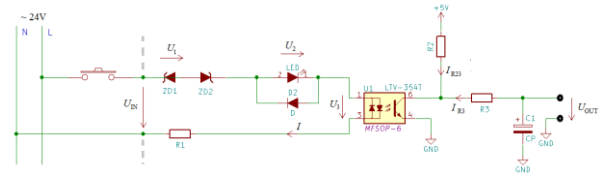
čo je menej ako jedna polperioda.

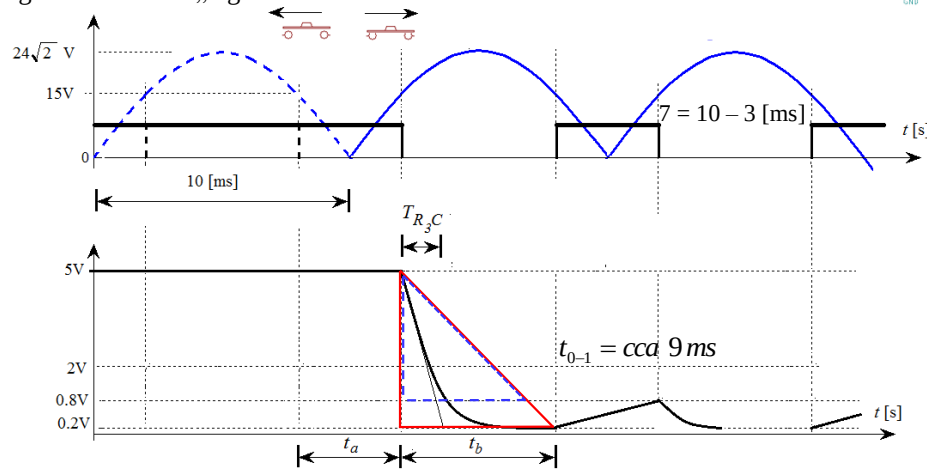
22

Niečo počítame, niečo kontrolujem. Cieľ je aby filter „reagoval/nereagoval“ počas jednej polperiódy kmitov.

Príklad:
Kontakt zopneme.

Za aký čas prejde U_{OUT} zo stavu „log. 1“ do stavu „log. 0“ ?



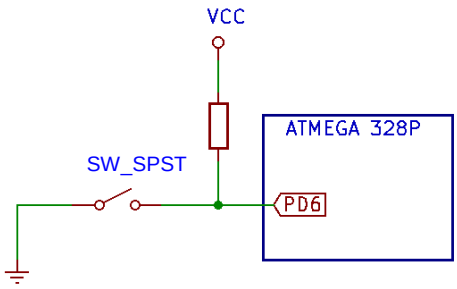


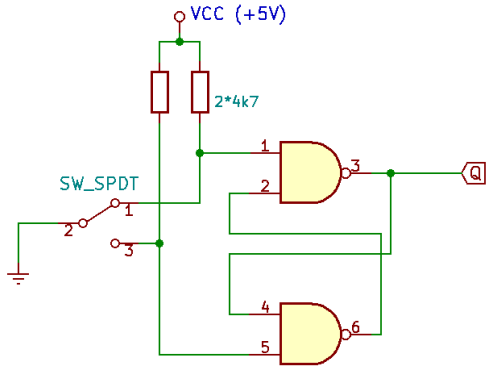
Pomer $\frac{4.8V}{4.2V} \approx 1 \Rightarrow$ musíme zobrať niekoľko časových konštánt, napr. **tri**.

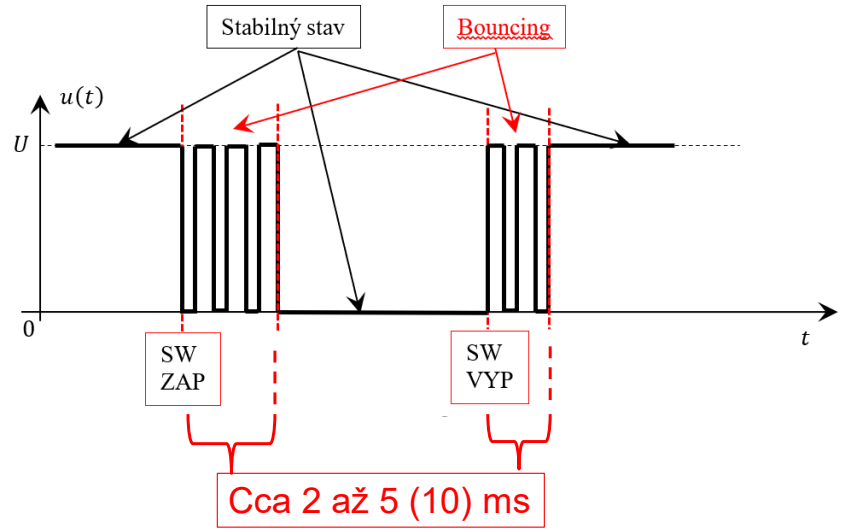
$\frac{4.8V}{4.2V} = \frac{3.T_{R_3C}}{7\,ms} \Rightarrow T_{R_3C} = 2.7ms$ Ak $R_3 = 2.7k\Omega$ Potom $R_2 \cong 22k\Omega$

23

Predchádzajúci výpočet vyzerá bezchybne. Treba ale pripomenúť, že tento návrh sme realizovali v čase TTL kompatibilný. Dnes máme gro obvodov CMOS. Z toho vyplýva, zase máme čo počítať.





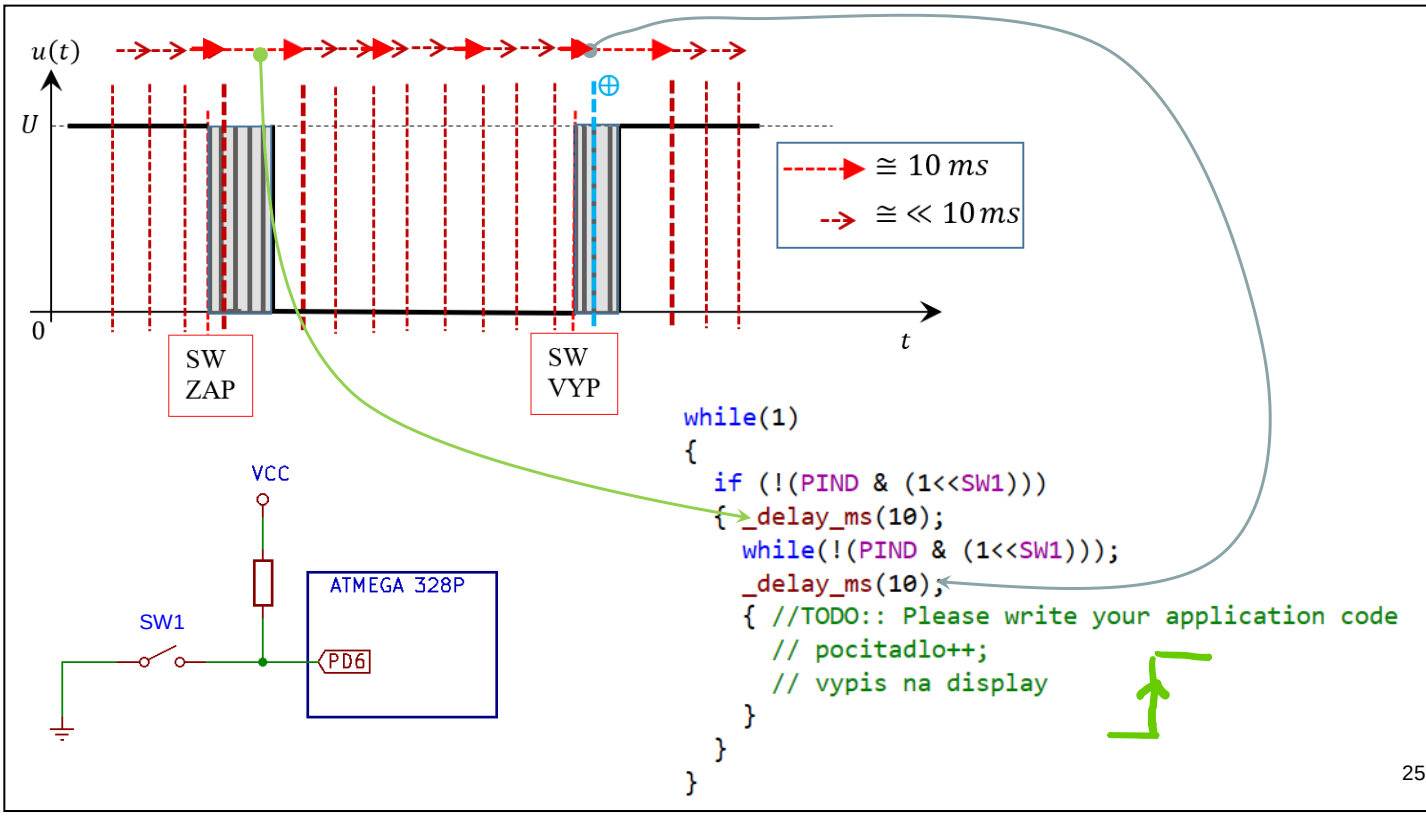


Ak nemáme softvérom vyhodnotiť viacnásobné zatlačenie, resp. uvolnenie tlačítka, musíme po detekovaní stavu tlačítka pár milisekund počkať až sa poloha kontaktu ustáli.

24

Dalo by sa povedať, že sme naznačili riešenie všetkých problémov ktoré sú spojené s „kontaktom“.

Zabudli sme na tzv. bouncing „odskakovanie“ - kmitanie kontaktu pri zopnutí, resp. pri rozopnutí. Možno si spomeniete, že na ZP sme rozprávali o RS preklápacom obvode. RS klopný obvod je takmer ideálne riešenie. Jeho nedostatkom je to, že predpokladá vypínač typu: SPDT = **S**ingle **P**ole, **D**ouble **T**hrow a náš kontakt je typu SPST = **S**ingle **P**ole, **S**ingle **T**hrow.



Úloha je jednoduchá. OPAKOVANE (v nekonečnej slučke) po každom zopnutí kontaktu vykonať jeden krát niečo (Napríklad, zmeniť stav LED-ky alebo inkrementovať **pocítadlo**.) a poprípade trochu neskôr ešte niečo iné (prijať/odoslať informáciu po sériovej linke).

Problémom úlohy s kontaktom je to, že AVR je rýchle. Ako naznačuje obr. počas stabilnej úrovne LOW by mohol program zbehnúť niekoľko krát.

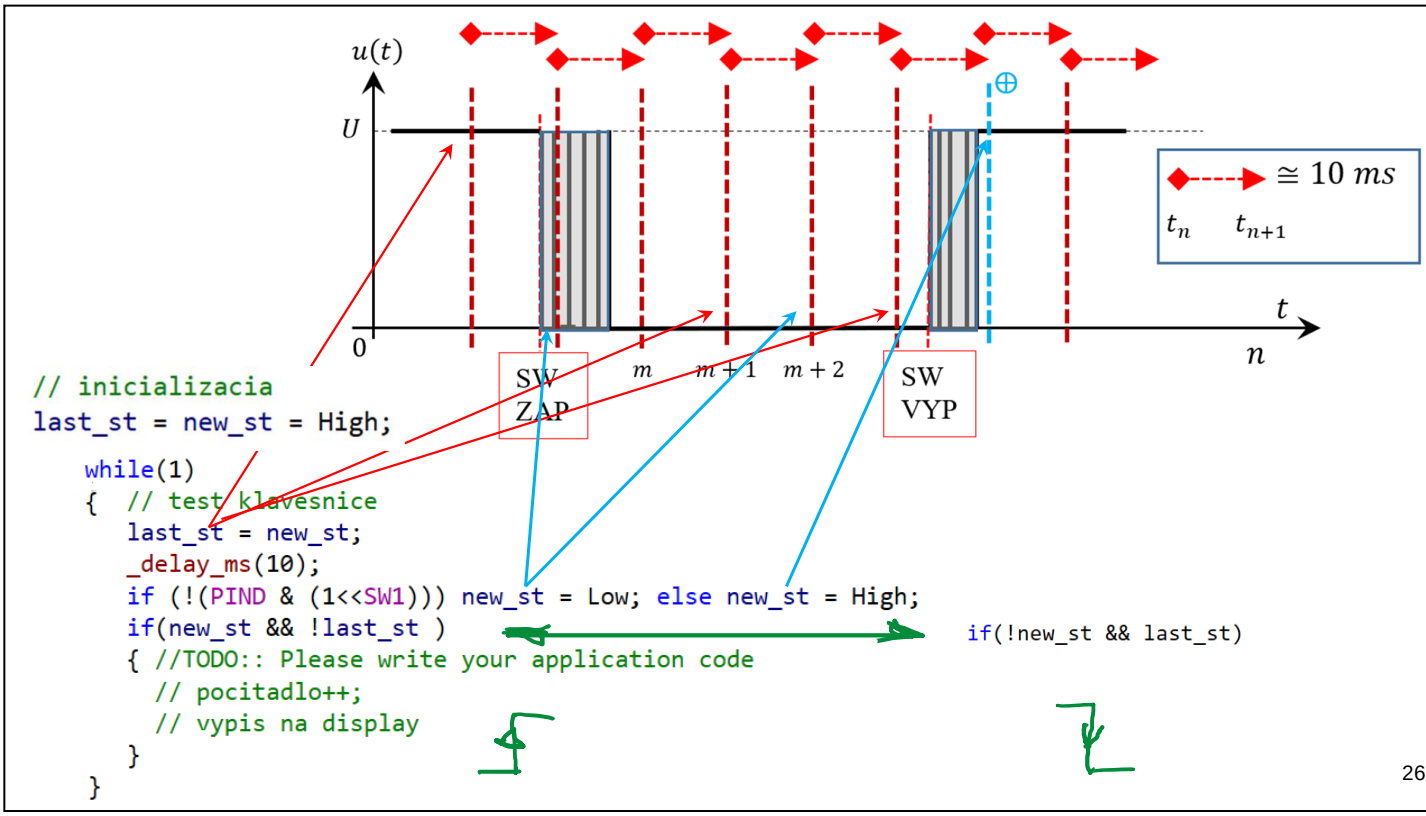
Vzorkovanie, ako sa ešte len budeme učiť, sa realizuje:

- v amplitúde. Pri kontakte má len dve úrovne. Log.1 (High) a Log. 0 (Low).
- v čase. V diskrétnom čase. Čas v MMP je odvodený od frekvencie oscilátora, od SC. Ešte nevieme nič o C/T a prerušovacom podsystéme, tak bude generovať oneskorenie pomocou funkcie delay(x ms), resp. len vykonáme niekoľko inštrukcií. Ak "zaregistrujeme" dobežnú/nábežnú hranu, počkáme cca 10ms, aby sme nevzorkovali opakované kmitanie kontaktu.

Ešte raz pripomeňme: systém reálneho času pracuje v nekonečnej slučke: while(True). Ako je zrejmé z priloženého programu, tento nedokáže realizovať dve činnosti (počítač je sériovo pracujúci stroj). Ak tlačítko zatlačíme a podržíme, bude do „nekonečna" stáť.

Ďalšou nevýhodou riešenia pomocou delay( ) je to, že počas tejto funkcie nemôže MMP realizovať iné činnosti.

POZNAMENAJME: Ak presunieme { // TODO: ... } za prvú funkciu _delay_ms( ) vykonáme niečo pri DOBEŽNEJ hrane.



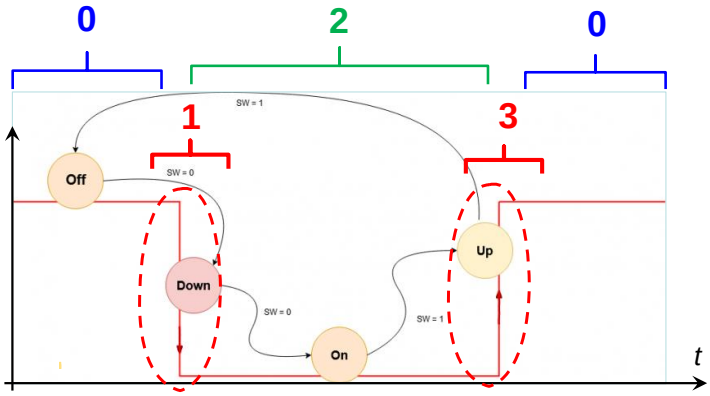
26

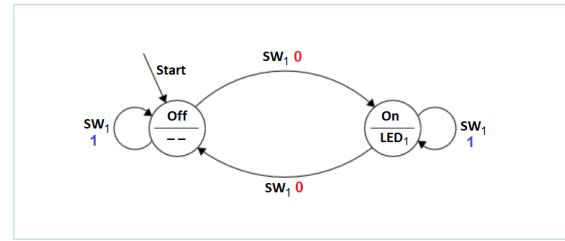
Nábežnú hranu môžeme zaregistrovať tak, že porovnáme stav kontaktu v predchádzajúcej a súčasnej vzorke. Iným spôsobom ako ošetriť kontakt proti zákmitom, je „zaregistrovať“ zmenu a potom sa po čase spýtať: Naozaj sa zmenil stav kontaktu?

Takto nejako pracuje aj prerušovací podsystém. Tá časť, kde sa vyhodnocuje nábežná, dobežná, resp. jednoducho hrana.

Cieľom cvičenia bolo:

Naprogramovať zapínanie a vypínanie led jedným tlačítkom.



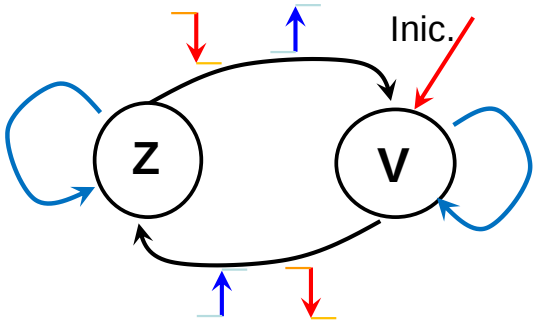


27

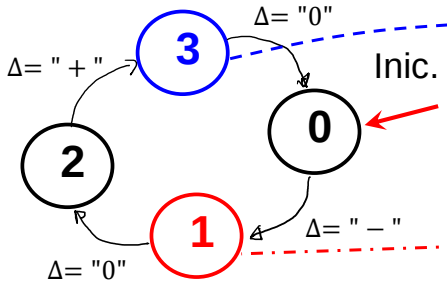
Stav ledky sa zmení pri zatlačení (1), niekde v strede (2) alebo pri pustení tlačítka (3)? Ak sa zhodneme, že zmena stavu sa udeje v (1), resp. (3) potom je namieste otázka. Kde a akým spôsobom treba pozmeniť vzorový program tak, aby sa stav ledky zmenil len raz pri „zatlačení – pustení tlačítka?“

Na prednáške sme na ošetrenie zákmitov používame „diferenciu“. Môžeme povedať, že príklad z prednášky je stavový automat a naopak príklad z cvičenia vyhodnocuje diferenciu, teda smer? Down, resp. Up?

LEDKA: `if ("TRUE") toggle_bit(PORTB,LED1);`



TLAČÍTKO: $\Delta = y(t_{n+1}) - y(t_n)$




```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#define set_bit(ADDRESS,BIT) (ADDRESS |= (1<<BIT))
#define toggle_bit(ADDRESS,BIT) (ADDRESS ^=(1<<BIT))
#define LED1   PB5          // zabudovana dioda
#define SW1    PD6          // tlacitko
enum states { Low, High };
int main(void){
enum states last_st = High;
enum states new_st  = High;
set_bit(DDRB,LED1);        // set pin LED1 as output
set_bit(PORTD,SW1);        // pull-up resistor ON
while(1){
   delay_ms(10);                    // chvilu pockajme
  { last_st =  new_st;
   if (bit_is_clear(PIND,SW1)) new_st = Low; else new_st = High;
   // if (new_st && !last_st) toggle_bit(PORTB,LED1);
   if (!new_st && last_st) toggle_bit(PORTB,LED1);
  }
 } /* end of while loop */
 return(0);                 // sem nikdy neprideme
}
```

28

Zadefinujme dva stavy Ledky:

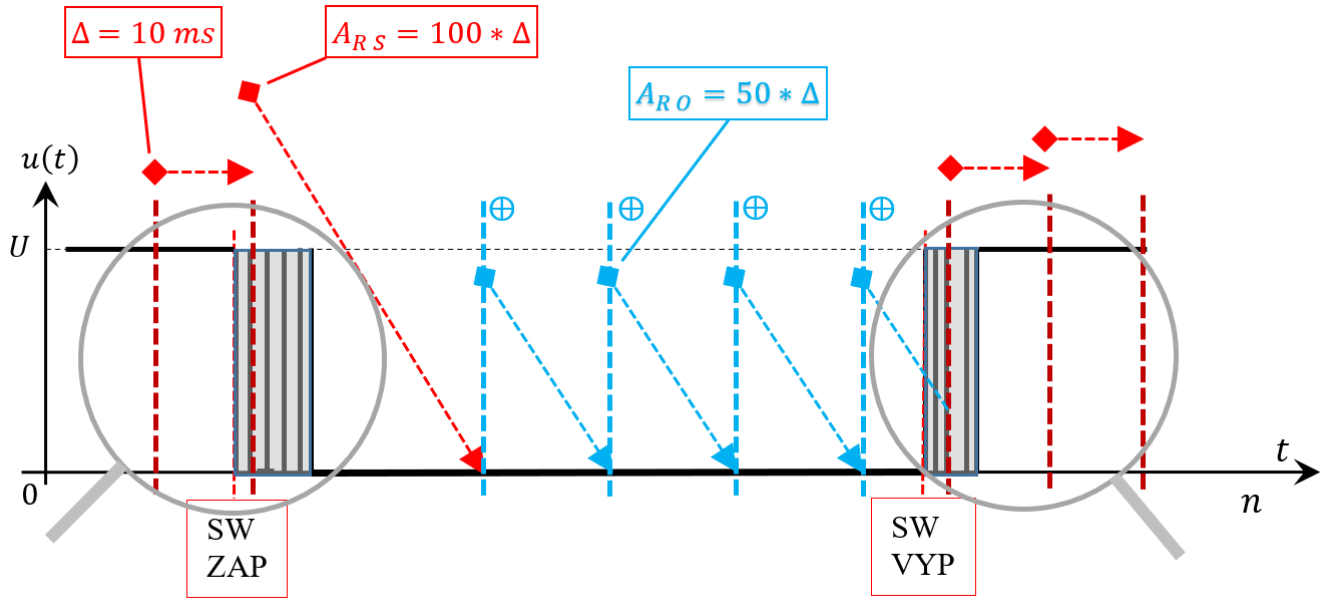
Z – zapnutá

V – vypnutá.

A prechod medzi týmito stavmi realizujme NÁBEŽNOU, resp. DOBEŽNOU hranou.

Predpokladáme, že po RESET-e je ledka vypnutá.

Autorepeat



29

Algoritmus Autorepeat funguje nasledovne: hlavná slučka zbieha každých 10 ms. Ak sa vyhodnotí prvá dobežná hrana prednastaví sa počítadlo A_{RS} na hodnotu, napr. 100, ktoré sa v hlavnej slučke každých 10ms dekrementuje. Po dopočítaní sa aktivuje rovnaká činnosť ako keby nastala nábežná hrana a prednastaví sa A_{RS} na hodnotu, napr. 50. Autorepeat sa vykoná skorej. Opäť ak dopočíta, aktivuje sa rovnaká činnosť ako keby nastala nábežná hrana. A celé sa to opakuje až do okamžiku nábežnej hrany. Teraz sa vykoná NIČ. Ak sa nábežná hrana objaví pred prvým dopočítaním počítadla, toto sa vynuluje a aktivuje sa činnosť odpovedajúca nábežnej hrane. Ak sa nábežná hrana objaví pri druhom, treťom, ... odpočítavaní, počítadlo sa vynuluje a vykoná sa NIČ.

Autorepeat:

```
// inicializacia
Int_AutRepeat = 0;  // interval Autorepeat
last_st = 1;        // kontakt rozopnuty
new_st = 1;         // kontakt rozopnuty
auto_repeat = 0;    // pocitadlo pre autorepeat
```

```
while(1)
{
    _delay_ms(10);
    // test klavesnice
    last_st = new_st;
    if(auto_repeat)auto_repeat--;
    if (!(PIND & (1<<SW1))) new_st = 0; else new_st = 1;
    if(new_st && !last_st && Int_AutRepeat)
    {   Int_AutRepeat = 0;
        auto_repeat = 0;
        { //TODO:: Please write your application code
          // pocitadlo++;
          // vypis na display
        }
    }
    if(!new_st && last_st){auto_repeat = 100; // 100*10 = 1sek
        Int_AutRepeat = 1;
    }
    if(!auto_repeat && !new_st && !last_st ){auto_repeat = 50;//    50*10 = 0,5sek
        Int_AutRepeat = 0;
        { //TODO:: Please write your application code
          // pocitadlo++;
          // vypis na display
        }
    }
}
```

30

Tento program sa voči predchádzajúcim líši v nasledovnom: Je tu náznak stavového automatu a oneskorenie (čas trvania slučky) je 10ms + „niekoľko“ SC.

Ak by sme 10ms –vé vzorky generovali cez prerušenie a „modro označenú časť programu“ vykonali len v čase 10 ms-vej vzorky slučka while(True) by trvala len niekoľko us alebo menej. Podľa toho či je prebeh cez slučku v čase 10ms vzorky alebo nie.

Ak budete niekedy aplikovať tento algoritmus, nezabudnite na: Bol optimalizovaný aj na počet inštrukcií aj na trvanie v čase. („Neprehadzujte poradie“.)

Digitálny potenciometer - KY-040 rotary encoder:

Handson Technology

Data Specs

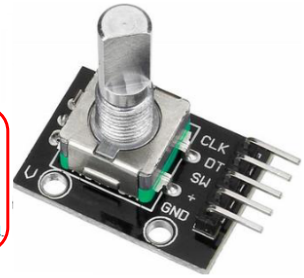
Rotary Encoder for Arduino/Raspberry

The KY-040 rotary encoder is a rotary input device (as in knob) that provides an indication of how much the knob has been rotated AND what direction it is rotating in. It's a great device for stepper and servo motor control. You could also use it to control devices like digital potentiometers.

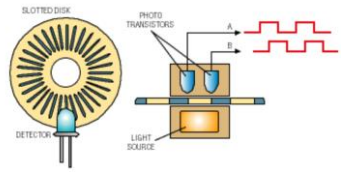
SKU: ASS-1058

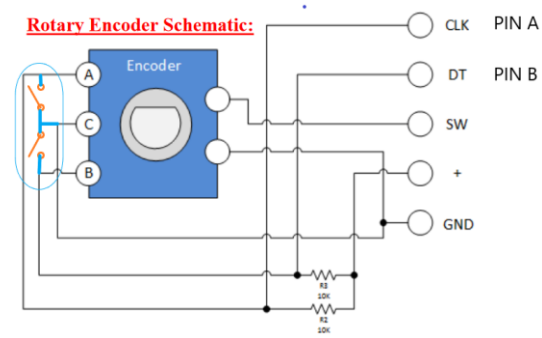
Brief Data:

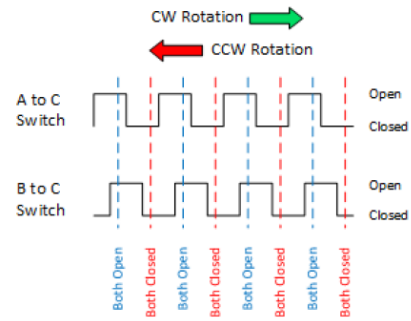
- Operating voltage: 5V.
- Pulses/360° Rotation: 20.
- Output: 2-bit gray code
- Mechanical Angle: 360° continuous.
- With built in push button switch (push to operate)
- Dimensions: (30 x 18 x 30) mm.
- Compatible with Arduino/Raspberry Pi controller board.



1 www.handsontec.com

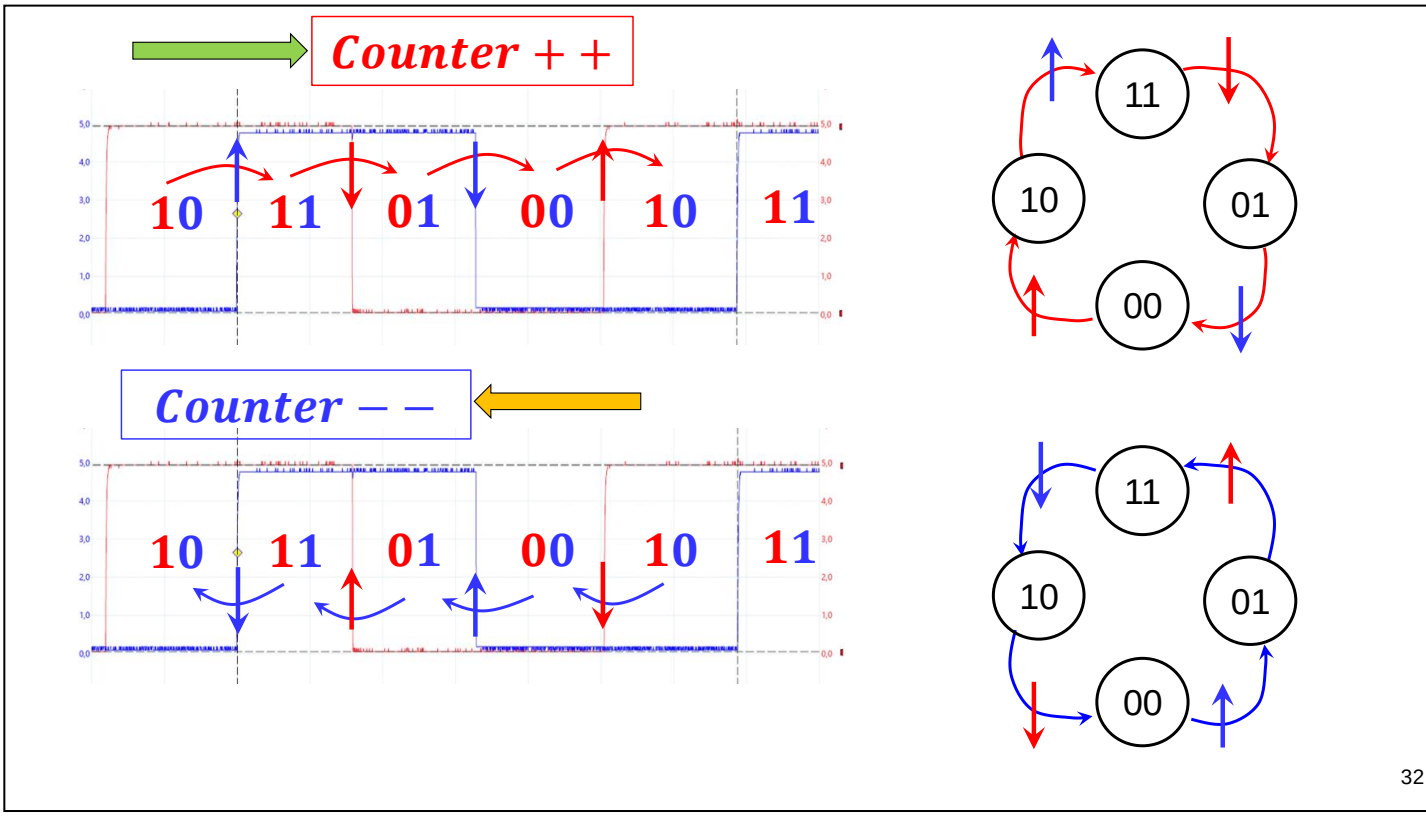






31

Ak zapojíme dva kontakty voči sebe posunuté (alebo niečo čo generuje informáciu log. 1 a log. 0) na niečo čo sa otáča, a túto informáciu vhodne spracujeme, budeme môcť odmeriavať polohu aj rýchlosť. A dokonca budeme vedieť určiť aj znamienko – teda smer.

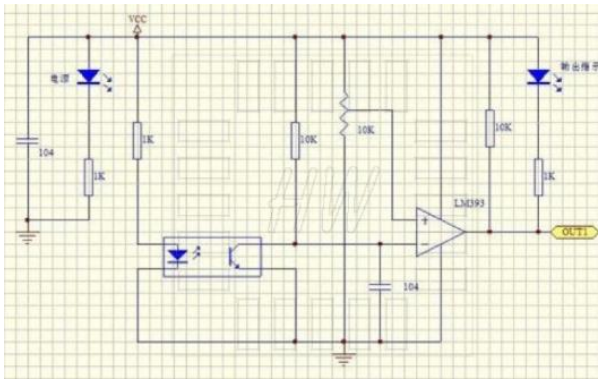


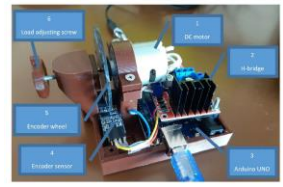
Ak sa budeme na rotačný encoder pozerať ako na stavový automat zo 4 stavmi 10 11 01 00, dokážeme softwarovo ošetriť kmitanie kontaktov. Ale to si budeme musieť na predchádzajúcej fólii „Brief Data“ všimnúť malú nepresnosť.

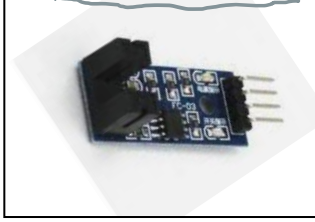
Keď sa budeme snažiť priradiť stavom (binárnym číslam) 10 11 01 00 dekadické 2, 3, 1, 0 nepôjdu pekne po sebe. Na týchto binárnych reťazcoch - číslach je zaujímavé to, že ich Hammingová vzdialenosť je JEDNA.

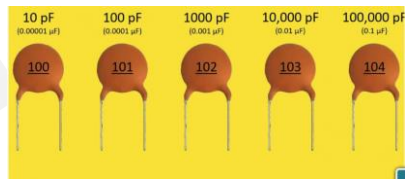
Je zaujímavé, že to čo sme robili „mechanicky“ (RC člen) potom „softwarom“ (oneskorenie) môžeme teraz urobiť len tak – použijeme vedomosti. Jednoducho povieme ono to bude fungovať správne.

Niečo podobné budeme robiť pri „generovaní“ predpätia pre display. Podľa KL toto napätie generujeme pomocou potenciometra. Potenciometer nahradíme OZ a RC filtrom. A nakoniec odstránime filter. Filtrom bude „nedokonalé“ oko.

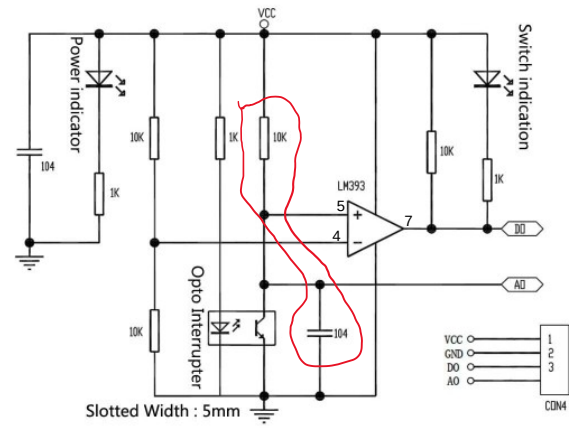




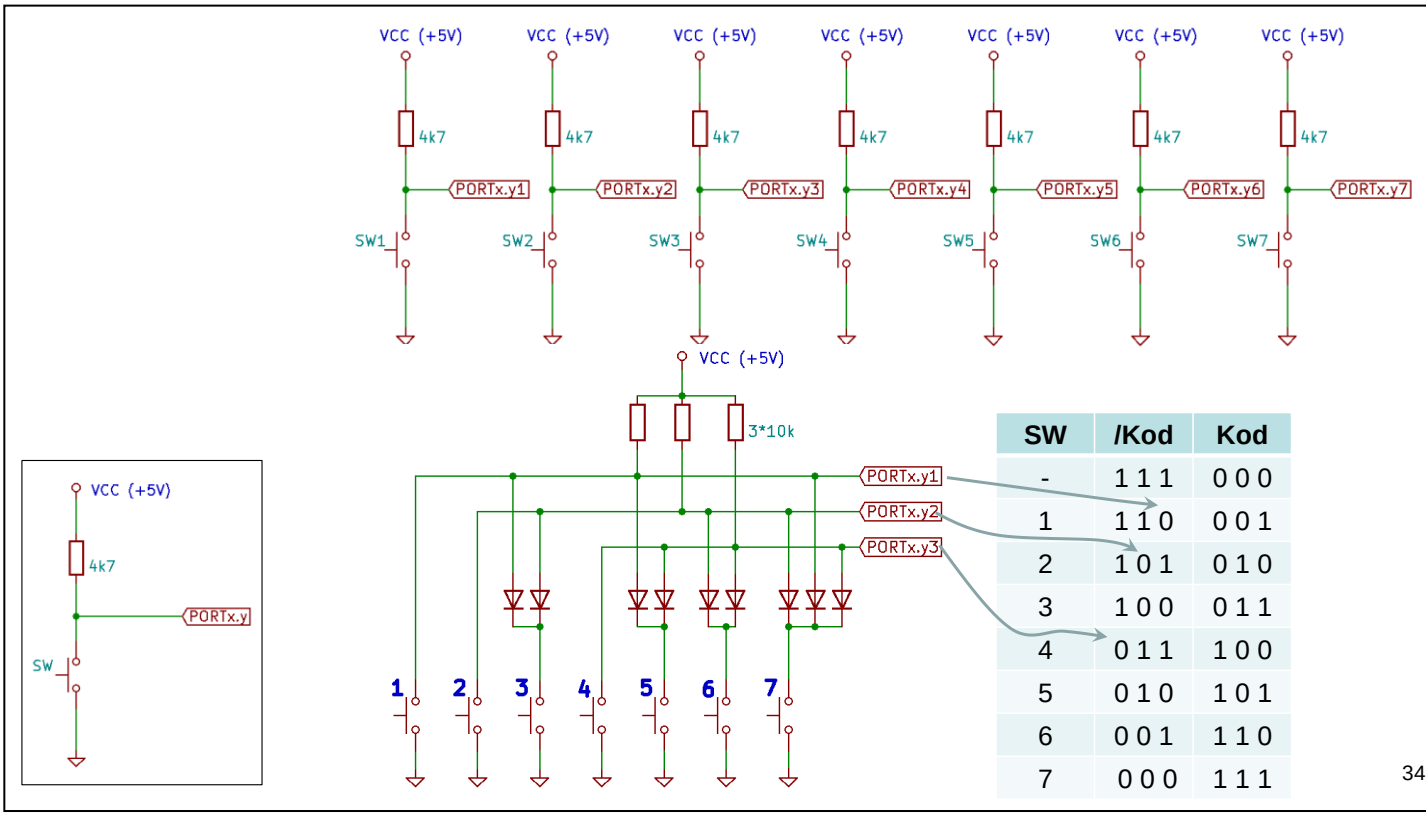




Module Circuit Diagram



33



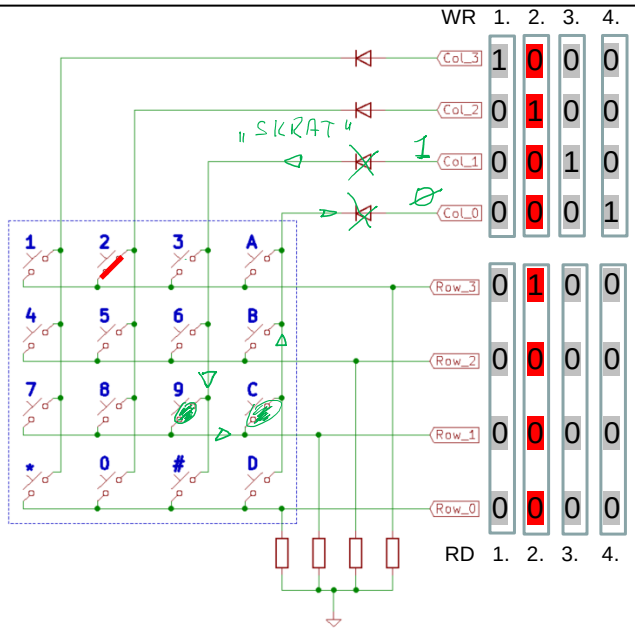
SW	/Kod	Kod
-	1 1 1	0 0 0
1	1 1 0	0 0 1
2	1 0 1	0 1 0
3	1 0 0	0 1 1
4	0 1 1	1 0 0
5	0 1 0	1 0 1
6	0 0 1	1 1 0
7	0 0 0	1 1 1

34

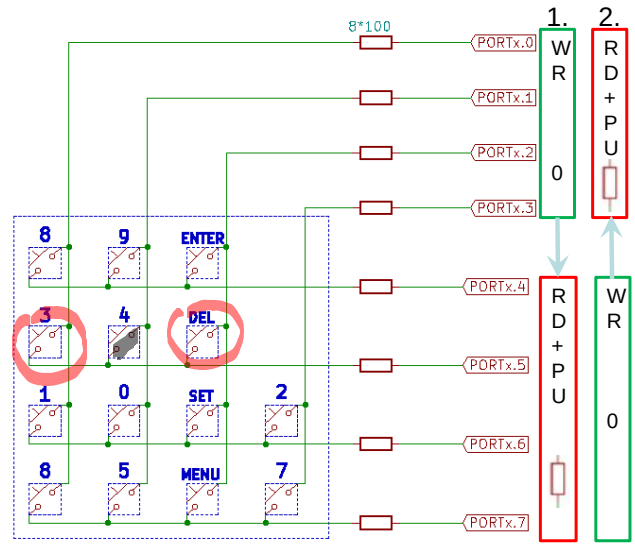
Pôvodne sme začali jedným kontaktom. Ak chceme realizovať viacero binárnych vstupov, lineárne bude narastať aj počet obsadených pinov, portov MMP.

Jedným z možných riešení je priradiť kód k tlačítku. Na 7 tlačítok potrebujeme 3 bity portu.

Nevýhodou tohto riešenia je , že nevieme vyhodnotiť súčasné zatlačenie tlačítok. Ďalšou nevýhodou, je vysoká úroveň log. 0 až 0,7V.



Klasický kontakt má prechodový odpor malý - 1-ky Ohmov a menej.



Fóliové klávesnice majú prechodový odpor zopnutého kontaktu cca 200 Ohm.

35

Obr. vľavo: Spotrebu počtu pinov portu (-ov) môžeme redukovať zapojením tlačítok do matice. Na 16 tlačítok nám postačuje 8 bitov portu. Ak chceme vyhodnocovať aj súčasné zatlačenie dvoch tlačítok, treba použiť naviac 4 diódy. Vyhodnocovanie sa robí tak, že postupne privádzame na jednotlivé stĺpce log. 1 a čítame odpovedajúci stav riadkov. Tento spôsob vyhodnocovania predpokladá, že na ostatné stĺpce privádzame log. 0. Viete uviesť konfiguráciu pinov, aby sme nemuseli použiť diódy? Vyhodnotenie je na 4 kroky.

Obr. vpravo: Tento spôsob vyhodnotenia zatlačených tlačítok nepredpokladá použitie diód. Naviac sme odstránili pulldown rezistory. Odporúčame ale použiť na vstupoch malé, cca 100Ohm-ové rezistory (obmedzenie prúdu).Vopred sme vylúčili také kombinácie súčasného zatlačenia tlačítok, ktoré sú na akejkolvek diagonále. Vyhodnotenie sa realizuje na dva kroky. Dokážete na dve „čítania “ zistiť zatlačenie ktoréhokoľvek tlačítka a väčšiny dvoch súčasne zatlačených?

Aj o tomto sú uP a ich programovanie.