

```
+----cursor parking lot----+
|                                |
|                                |
+-----+
```



Monnega Good, 2021 colorized --->



IMP - priprava na pulsemestralni zkousku (Jany doc)

Obsah:

[Sekce A: Pulsemka 2020 - skupina WIS](#)

[Sekce B: Pulsemka 2017 - skupina A](#)

[Sekce C: Pulsemka 2016 - skupina B](#)

[Sekce D: Pulsemka 2015 - skupina A](#)

[Sekce E: Pulsemka 2019 - skupina A](#)

[Sekce F: Pulsemka 2021 - skupina A](#)

[Sekce G: Pulsemka 2021 - skupina B](#)

Dalsi uzitecne linky:

Mazanec explains:

<https://www.youtube.com/playlist?list=PLsqIxixRzWlw1H0SmfeTxPnWvPMqaMlUQc>

DealerDify stream:

<https://www.youtube.com/watch?v=1rhvi8mrHbQ&t=107s>

Prednášky 2021:

https://youtube.com/playlist?list=PL_eb8wrKJwYuRHpNm6eGAZX1P65hvDxEu

Zpetna vazba od Strnadela/Ruzicky k pulsemce rok 2016 (sekce C):

<https://fituska.eu/download/file.php?id=12589&sid=e897ec5667a066b0f12c3ace0a03d31e>

Zpetna vazba od Strnadela/Ruzicky k pulsemce rok 2017 (sekce B):

https://drive.google.com/file/d/1Cx8LX7TBGd5GiQOSVmwg4pcI1_BByAc/view

IMP testy a zapisky (z laboratorních cvičení)

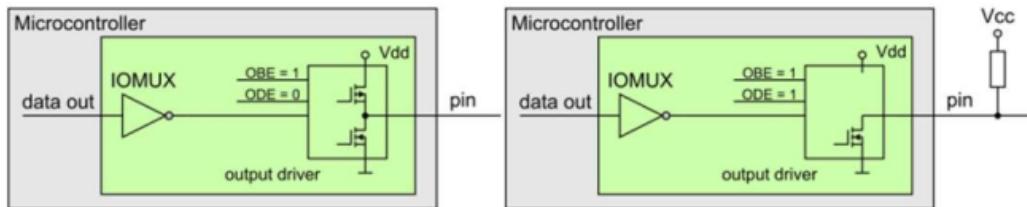
https://docs.google.com/document/d/1SW3rynyz_IdkXK9frk1LDNtPJ_CXb-zaeVvem9j5Cik/edit#

Q:

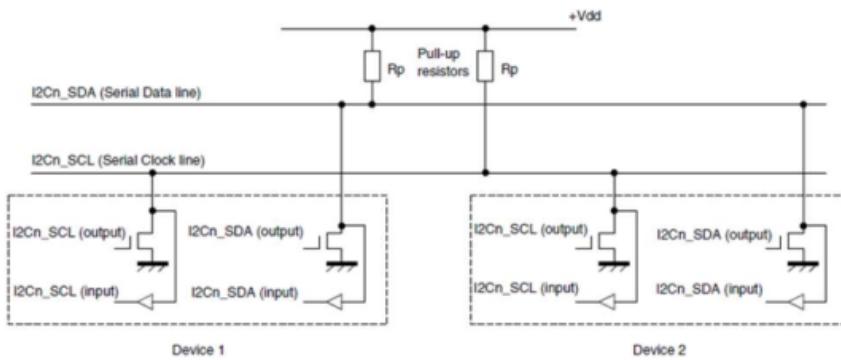
Popište, jakou funkci mají otevřené kolektory (nebo open drain výstupy) v koncových budičích sběrnice u I2C
- proč se tam používají.

Otevřený „kolektor“

(přesněji: „open drain“)



Typické použití: buzení sběrnice více zdroji – aby nedocházelo ke zkratům:



25

Timestamp v prednasce: <https://youtu.be/i1KDPWHgYQE?t=5747>

Q:

Popište stručně, k čemu se používá tzv. **Link register (R14)** u mikrokontrolérů s jádrem ARM Cortex M0+ a co obsahuje (jaký typ informace je v něm uložen)?

LR obsahuje adresu inštrukcie (hodnotu programového čítača), z ktorej má program po vrátení z funkcie pokračovať. Programový čítač načíta hodnotu z tohto registra po tom, ako bola funkcia ukončená.

funkcí může být i nějaká subrutina (obsluha přerušení atp.)

Q:

Pulsně-šířková modulace (PWM) se používá

Zvolte správnou odpověď:

- k odstranění zákmitů mechanických kontaktů (například tlačítek)
- jako způsob, jak realizovat pseudo-analogový výstup na číslicovém výstupním pinu
- pro periodické generování přerušení

spravna odpoved je ta prostredni

Q:

Napište příkaz v C, kterým potvrďte přerušení od pinu 5 portu B v registru PORTB_ISF.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

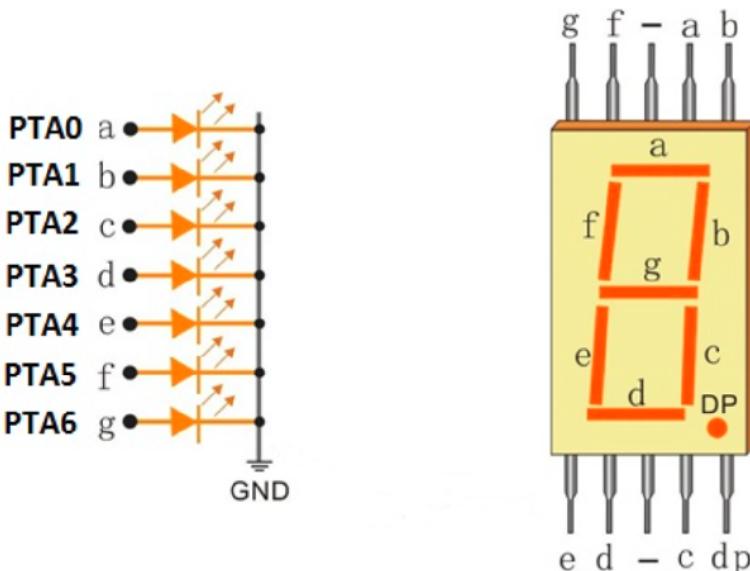
PORTx_ISFR field descriptions

Field	Description
31-0 ISF	Interrupt Status Flag Each bit in the field indicates the detection of the configured interrupt of the same number as the field. 0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic one is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

```
PORTB->ISFR = 0x00000020; // prime nastaveni registru
PORTB->ISFR &= (1 << 5); // nebo take tohle (coz je mozna jeste korektnesji)
- zakladem je, aby se na 1 nastavil jenom ten pin, ktereho preruseni chceme
potvrdit
- kdyby se treba pouzitim OR operace ponechal zvysek konfigurace registru, tedy
by zustalo treba vice pinu nastaveno na 1, tak se po obsluze jednoho (toto
naseho) preruseni vsechny tyto bity vynuluji a tim se ztrati informace o jejich
zadosti o preruseni
```

Q:

Napište příkaz v C pro rozsvícení číslice 5 na displeji. Displej je připojen k portu A mikrokontroléru dle schématu.

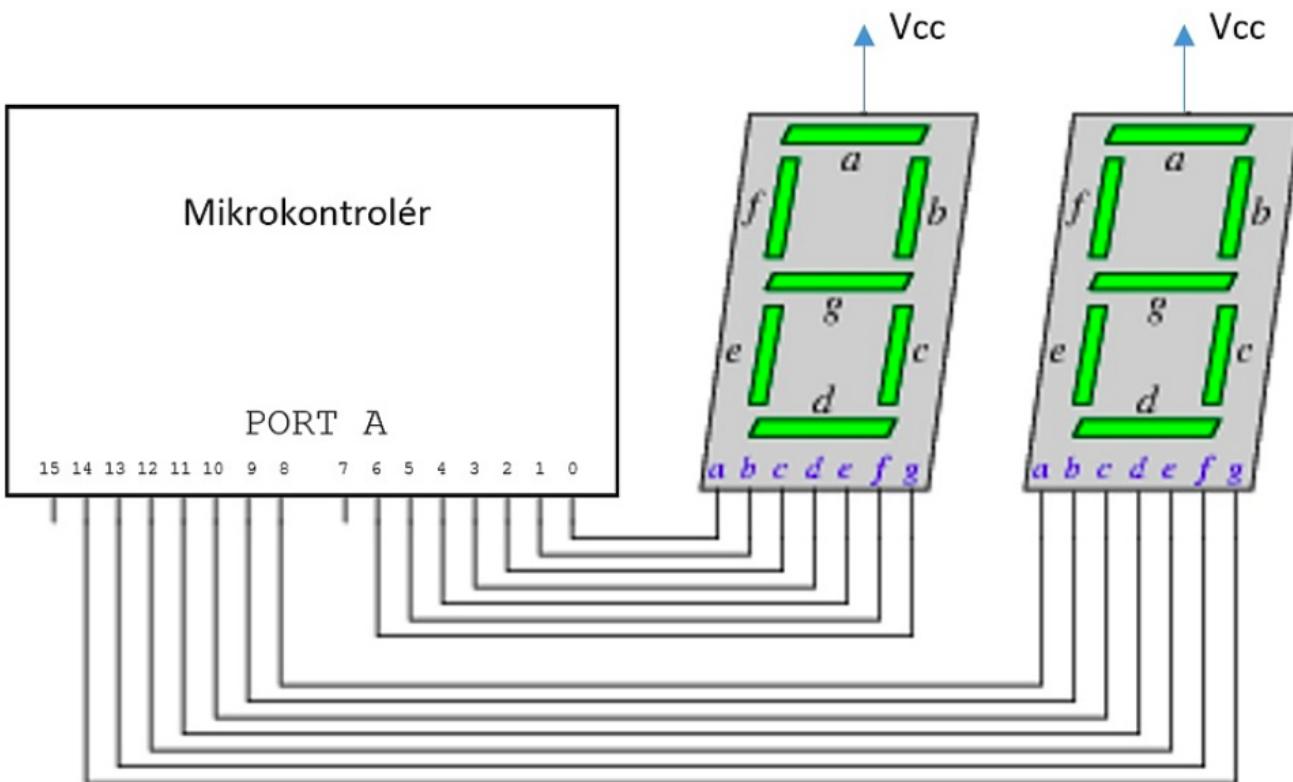


protoze mame piny pripojeny pri anode (schema diody = --- anoda >| katoda ---), a pri katode je GND, tak chceme pro rozsviceni pozadovanych segmentu dostat na piny logicku jednicky, a tedy:

```
// korespondujici segmenty: 0bPgfedcba // P = (decimal) point
PTA->PDOR = 0x6D // 0x6D = 0b01101101
```

Q:

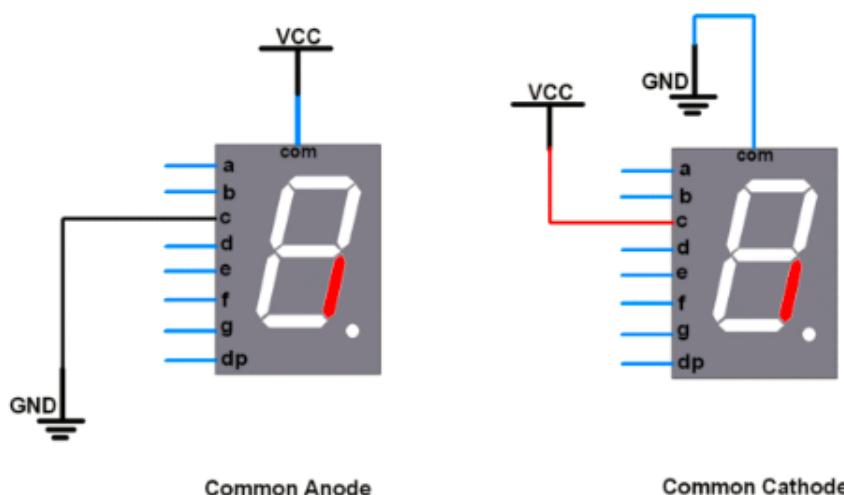
Na dvou sedmisegmentových displejích, připojených k mikrokontroléru dle obrázku, svítí nějaké dvě číslice. **Napište příkaz v C, kterým změníte číslo na displeji jednotek** (displej vpravo) **na 7, aniž byste ovlivnili** to, co je zobrazeno na displeji desítek. Povšimněte si, že LED segmentů displeje mají společnou anodu (viz připojení na Vcc nahoře)!



protoze mame nahore zapojene napeti (je pripojene k anode a piny jsou pri katode), pak potrebueme dostat nuly na piny tech LEDek, ktere chceme rozsvitit, abychom dostali rozdil napeti potrebny pro rozsviceni, tedy:

```
PTA->PDOR |= 0x7F00 // zhasnuti vsech segmentu na 2. displeji  
PTA->PDOR &= 0x787F // rozsviceni "7" na 2. displeji
```

```
// jelikoz piny 15 a 7 nejsou nikam zapojeny, pak v reseni mozno ve vsech hex  
// hodnotach nahradit 7 za F a bude to porad OK  
// nekdy ale ty piny byvaji zapojeny do LEDky co ukazuje tecku za cislem, na to pozor
```

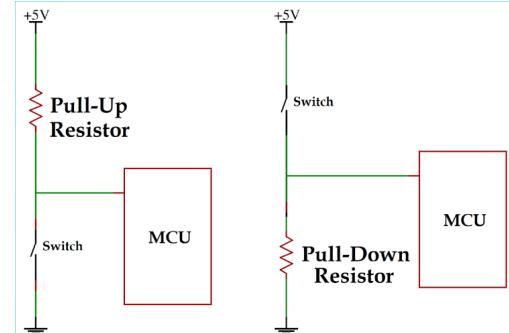
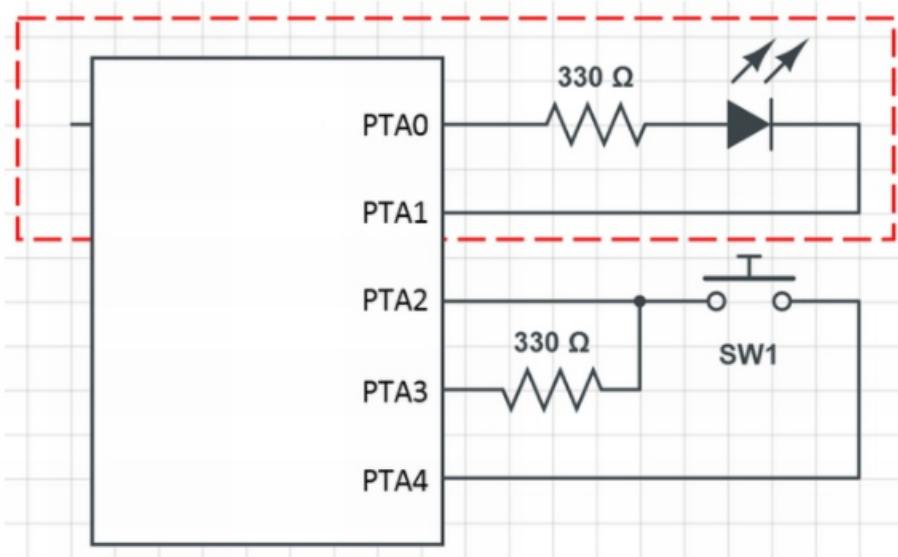


```
// nazorny obrazek pro vysvetleni rozdilu tohohle prikladu a toho predesleho  
// obrazek vlevo je tenhle priklad, obrazek vpravo je predesly priklad (s tou 5)  
Timestamp k prednasce, kde se vysvetluje ovladani 7segmentoveho displeje:
```

https://youtu.be/i1KDPWHgYQE?list=PL_eb8wrKJwYuRHpNm6eGAZX1P65hvDxEu&t=1901

Q:

Na port A mikrokontroléru MKL05Z jsou připojeny součástky dle schématu. Určete hexadecimálně obsah registrů GPIOA_PDDR a GPIOA_PDOR tak, aby LED svítila a stav tlačítka bylo možno snímat. Předpokládejte, že ostatní piny portu A nebudou využity.



Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	PDD	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
W	1																1																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			

GPIOx_PDDR field descriptions

Field	Description
31-0 PDD	Port Data Direction Configures individual port pins for input or output. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function.

PDOR je output register (pro poslání)

PDIR je input register (pro čtení)

PDDR nastavuje, jaký pin je input (log.0) a jaký output (log.1)

- PTA3 (ten s rezistorem) jde (měl by jit) do země a funguje jako pulldown resistor
- PTA4 je tím pádem VCC, takže když zmáčkneš tlačítko, na vstupu bude 1, jinak to rezistor (pta3) stáhne k 0
- PTA2 je tedy input
- PTA0 je output, kterej můžeme nastavovat. Chcej aby LEDka svítila -> musí být tam VCC, takže 1
- PTA1 bude output nastavený na 0 (protože tam jde katoda té diody)

tedy: (jsou 2 různé správné konfigurace GPIOA_PDOR - pull-up a pull-down rezistor)

// PDDR = co je output a input, PDOR = co má být high a co low; vcc a gnd

GPIOA_PDDR = 0x0000001B // 0b0000000000000000000000000000000011011 -> pta0 je LSb

A) GPIOA_PDOR = 0x00000011 // 0b0000000000000000000000000000000010?01 -> otazníky do nul ???

// ^ tady se jedná o pull-down verzi rezistoru (koresponduje s popisem výše)

B) GPIOA_PDOR = 0x00000009 // 0b00000000000000000000000000000001?01 -> otazníky do nul ???

// ^ tohle je pull-up verze rezistoru, při sepnutém spinaci bude na PTA2 log. 0

// pomocný obrázek (pull-up/down res. nahore napravo) -> pin z MCU je v našem případě PTA2,

// když na PTA3 privědeme 1 a na PTA4 0, pak se bude jednat o pull-up, tedy při vypnutém

// spinaci tenhle rezistor přinese na PTA2 log.1 a při sepnutí se PTA2 prepojí se zemí (PTA4)

// opacně když PTA3 bude mít 0 a PTA4 log.1, tak se jedná o pull-down verzi

Q:

Tlačítko na vstupu mikrokontroléru potřebuje další součástku (rezistor) proto, aby

Zvolte správnou odpověď:

- byla definována na vstupu jasná úroveň při rozpojeném kontaktu tlačítka
- bylo možno udržet stejný proud při stisknutém i puštěném tlačítku
- se snížila spotřeba zařízení při stisku tlačítka

Q:

Hodnota, která je na výstupním pinu portu, lze ovlivnit

Zvolte správnou odpověď:

- pouze zápisem do registru PDOR (Port Data Output register)
- zápisem do registrů PDOR (Port Data Output Register), PTOR (Port Toggle Output Register), PSOR (Port Set Output Register) a PCOR (Port Clear Output Register)
- zápisem do registrů PDOR (Port Data Output Register) nebo PDIR (Port Data Input Register)

Q:

Čím se zapíná modul GPIO?

Zvolte správnou odpověď:

- Zapíná se povolením hodinového signálu pro modul.
- Zapíná se povolením napájení modulu.
- Je zapnutý vždy.

Q:

Délka přechodného děje při sepnutí nebo rozepnutí kontaktu tlačítka (zákmity) je asi

Zvolte správnou odpověď:

- několik mikrosekund
- několik milisekund
- několik nanosekund

- milisekund je spravne, je to z dokumentu IMP testy a zapisky (strana 4/9 dolu)

Q:

Piny na pouzdro mikrokontroléru, které je možné v registrech PORTx_PCRn přiřadit časovači TPM, jsou svázány

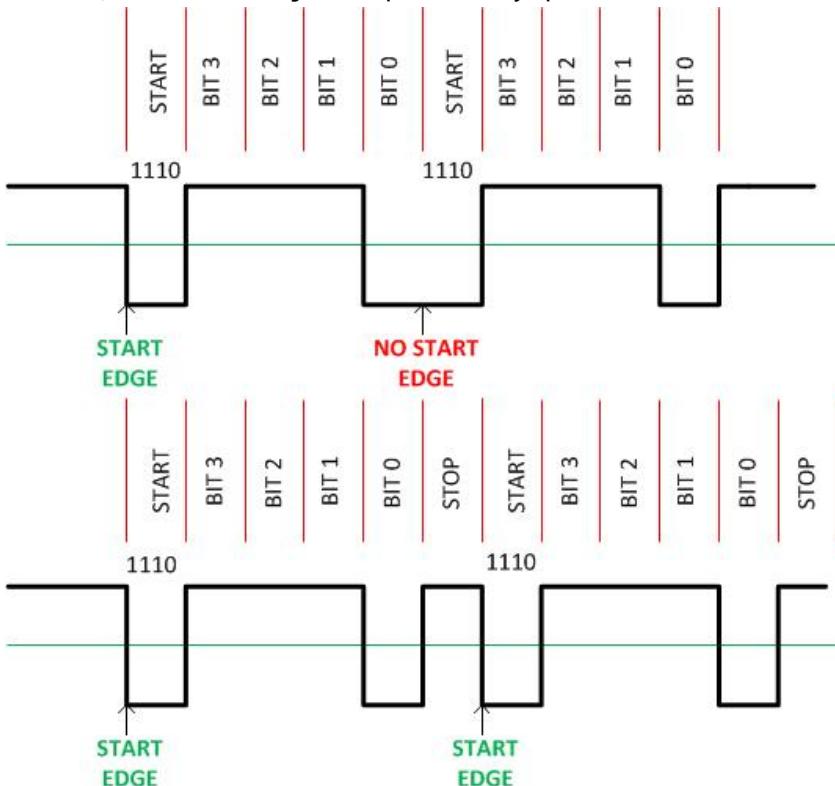
Zvolte správnou odpověď:

- s jednotlivými kanály časovače
- s přerušením od přetečení čítače
- s výstupy čítače v časovači

Q:

Popište, co by se mohlo špatného stát v případě, že u asynchronní sériové komunikace (modulem UART) by neexistoval žádny stop-bit a rámec by tak končil posledním datovým bitem.

Stop bit slouží k vzájemnému oddělení přenášených slov. Kdyby neexistoval, pak nevíme, kde končí jeden prenaseny paket a kde zacina druhý.



Q:

Popište, jakým způsobem si při sériové komunikaci standardem SPI mikrokontrolér jako master vybírá, s jakým modulem typu slave bude právě komunikovat.

Pomoci signalu SS (Slave Select), který slouží k vyberu, s kým master komunikuje.

Z prezentace:

Výběrový signál SS (Slave Select), který je aktivní v log. 0, slouží k výběru a aktivaci zařízení, se kterým má probíhat komunikace.

Na zařízení master je signál SS výstupem připojeným k modulu slave. Ten je možné prostřednictvím mastera signálem SS aktivovat a zahájit tak datový přenos.

Master musí zajistit, že SS bude po dobu přenosu v log. 0 (aktivní). V daném čase smí probíhat komunikace pouze s jediným modulem slave, ostatní zařízení, jejichž SS vstup je v log. 1 (neaktivní) do komunikace nezasahuje.

Q:

Co je to endianness? Proč je možné ji u mikrokontrolérů ARM Cortex nastavovat?

Endianness udává, v jakém pořadí jsou uloženy jednotlivé bajty číselného datového typu (který zabírá více než 1 bajt) do operační paměti.

Dá se nastavovat, aby nedocházelo k nedorozumění při komunikaci dvou systémů.

Big endian = MSB daného typu je na nejnižší adrese, LSB na nejvyšší

Little endian = LSB daného typu je na nejnižší adrese, MSB na nejvyšší

Q:

K čemu slouží standard MISRA-C (Motor Industry Software Reliability Association) a na jakém principu funguje?
Co je podstatou, díky které může svůj účel naplnit? Popište stručně.

Tento standard slouží především pro zlepšení bezpečnosti kódu, přenositelnosti a spolehlivosti v kontextu vestavěných systémů.

Podstatou tohohle standardu je množina pravidel, které jistým způsobem omezí programovací jazyk, aby se zamezilo vzniku chyb.

Q:

Konstrukce

```
if (x = y)  
{ foo(); }
```

je sice v jazyce C správně, ale **neodpovídá pravidlům standardu MISRA-C** (Motor Industry Software Reliability Association). **Popište stručně, proč.**

protože MISRA-C nepripousti používání přirazovacího operátora v logické operaci (podmínka)

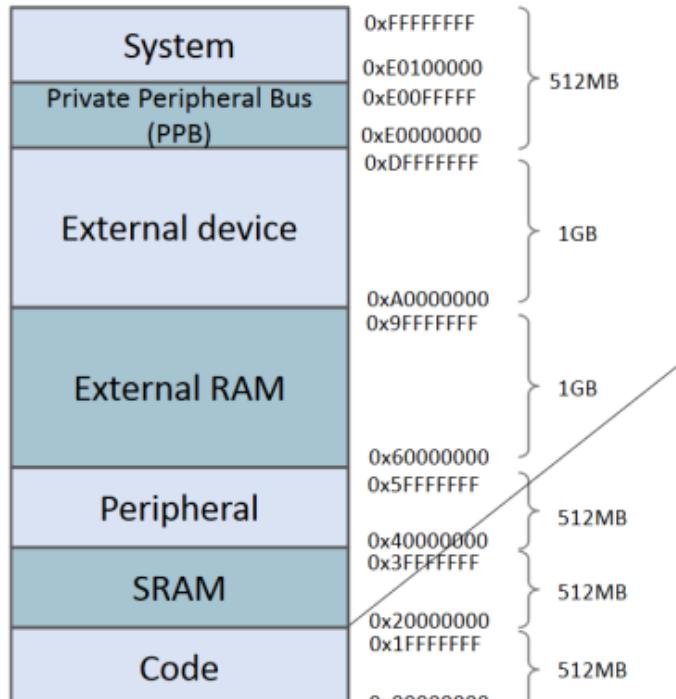
"Rule 13.1 (required): Přiřazovací příkaz nesmí být použit v logickém výrazu."

Q:

Popište (např. adresou nebo jinak specifikujte umístění v adresovém prostoru), **kde v adresovém prostoru mikrokontrolérů** s jádrem ARM Cortex M0+ **je prostor pro hlavní paměť typu Flash**, tj. paměť programu? **Jak velký je celý adresový prostor** mikrokontrolérů ARM Cortex M0+?

Cortex-M0+ Memory Map

- Reserved for other purposes
- Private peripherals
e.g. NVIC, SCS
- Mainly used for external peripherals
e.g. SD card
- Mainly used for external memories
e.g. external DDR, FLASH, LCD
- Mainly used for on-chip peripherals
e.g. AHB, APB peripherals
- Mainly used for data memory
e.g. on-chip SRAM, SDRAM
- Mainly used for program code
e.g. on-chip FLASH



Flash zabírá spodních 512 MiB adresového prostoru od adresy 0. Tvoří osminu celkové paměti; celý adresový prostor ARM Cortex M0+ má velikost 4 GiB.

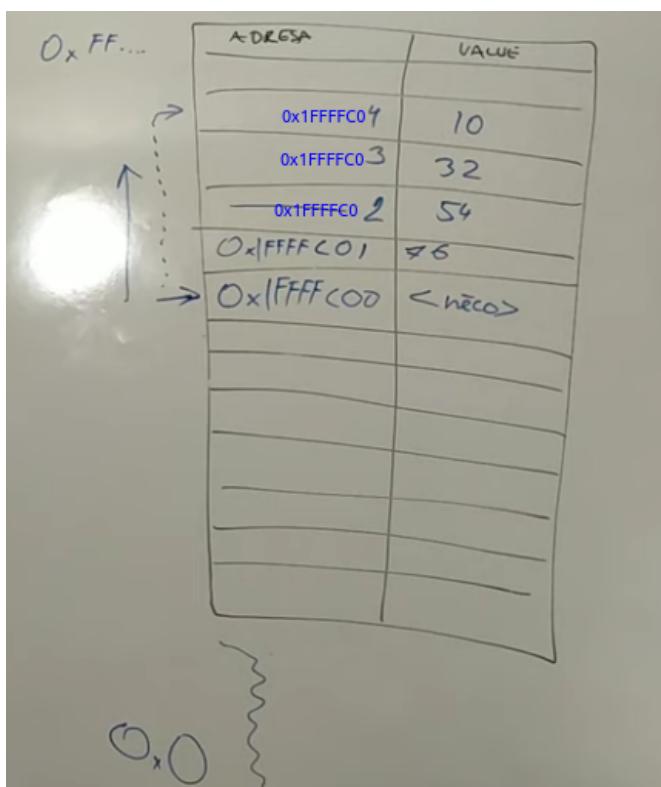
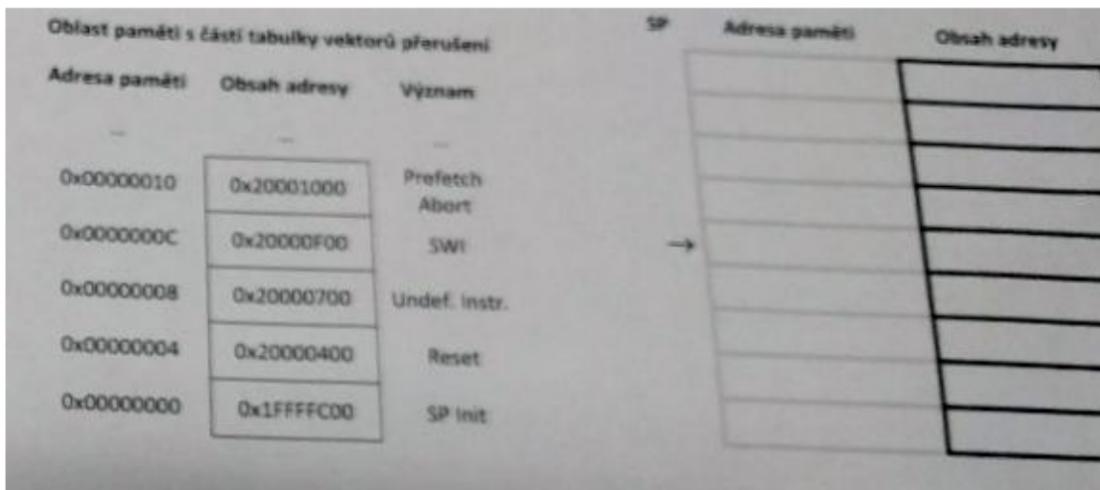
Sekce B: Pulsemka 2017 - skupina A

Q:

Předpokládejte, že

- bezprostředně po resetu MCU založeném na jádru ARM, s endianness formátu byte-invariant big-endian, bude na zásobník vložena hodnota 0x76 54 32 10
 - adresa paměti roste směrem nahoru, výchozí pozici SP vyobrazuje šipka

Do obrázku vpravo doplňte pozici SP, příslušné adresy a jejich obsahy tak, aby byl ilustrován stav bezprostředně po vložení hodnoty na zásobník (obsahy adres a paměťových buněk nedotčených vložením hodnoty na zásobník nevypĺňujte).



```
// ^^ v tomhle reseni se jednalo o big endian format, full ascending stack
// endianness jenom udava, jak budou ulozeny data v pameti
// big endian = MSB je na nejnazsi adrese, little endian = LSB je na nejnazsi adrese
// dale je dulezite, jestli se jedna o ascending nebo descending stack
// ascending po PUSH jede na vyssi adresu (s vyssi hodnotou), descending na nizsi
// dalsi parameter je, jestli se jedna o full nebo empty - full znamena, ze v klidove
// poloze ukazuje SP na posledni zaplnenou adresu, empty bude ukazovat na prvni
// volnou, teda treba empty pri push nejprv zaplni aktualni adresu a pak se pohnet SP,
// pri full by to bylo naopak (tedy nejprv zmenna SP a pak zaplneni dane adresy)
// je potreba se ridit zadanim, ale by default je ARM full-descending
```

Q:

Zadání: Doplňte funkci v jazyce C pro přijetí znaku z asynchronní sériové linky prostřednictvím modulu UART0 mikrokontroléra MKL05Z. Funkce má čekat na příchozí znak a ten pak vrátit. Předpokládejte, že modul UART0 je již kompletně inicializován a nastaven.

UART memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value
4006_A000	UART Baud Rate Register High (UART0_BDH)	8	R/W	00h
4006_A001	UART Baud Rate Register Low (UART0_BDL)	8	R/W	04h
4006_A002	UART Control Register 1 (UART0_C1)	8	R/W	00h
4006_A003	UART Control Register 2 (UART0_C2)	8	R/W	00h
4006_A004	UART Status Register 1 (UART0_S1)	8	R/W	C0h
4006_A005	UART Status Register 2 (UART0_S2)	8	R/W	00h
4006_A006	UART Control Register 3 (UART0_C3)	8	R/W	00h
4006_A007	UART Data Register (UART0_D)	8	R/W	00h

37.2.5 UART Status Register 1 (UARTx_S1)

Address: 4006_A000h base + 4h offset = 4006_A004h

Bit	7	6	5	4	3	2	1	0
Read	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
Write				w1c	w1c	w1c	w1c	w1c
Reset	1	1	0	0	0	0	0	0

5 RDRF	Receive Data Register Full Flag RDRF becomes set whenever the receive data buffer is full. To clear RDRF, read the UART data register (UART_D). 0 Receive data buffer empty. 1 Receive data buffer full.
-----------	--

// S1 je 8 bitový a příznak RDRF je na 6. (5) bitu což je $0b00100000 = 0x20$
// na cvičení se použilo UART0_S1_RDRF_MASK, které je v knihovně "MKL05Z4.h"

```
char ReceiveChar(void) {
    while (!(UART0->S1 & 0x20)); // místo 0x20 můžno taky použít (1 << 5)
    return UART0->D;
}
```

// SendChar by byl takhle

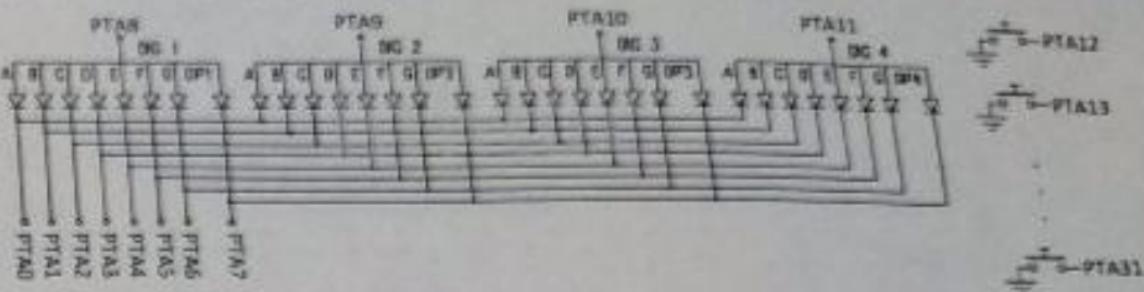
Bit	Description
7 TDRE	Transmit Data Register Empty Flag TDRE is set out of reset and whenever there is room to write data to the transmit data buffer. To clear TDRE, write to the UART data register (UART_D). 0 Transmit data buffer full. 1 Transmit data buffer empty.
6 TC	Transmission Complete Flag TC is set out of reset and when TDRE is set and no data, preamble, or break character is being transmitted. TC is cleared automatically by one of the following: <ul style="list-style-type: none">• Write to the UART data register (UART_D) to transmit new data• Queue a preamble by changing TE from 0 to 1• Queue a break character by writing 1 to UART_C2(SBK) 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).

```
// 0x80 = maska pro bit s TDRE flagem, 0x40 = maska pro bit s TC flagem
// místo 0x40 a 0x80 taky můžno použít (1 << 6) a (1 << 7)
void SendCh(char ch) {
    while (!(UART0->S1 & 0x80) && !(UART0->S1 & 0x40));
    UART0->D = ch;
}
```

Q:

Na port A mikrokontroléru MKL05Z je připojen čtyřmístný sedmisegmentový displej a tlačítka dle následujícího rozpisu a schématu:

PTA0 – PTA7 ... segmenty A-F a desetinná tečka DP (katody), PTA8 – PTA11 ... anody jednotlivých číslicovek DIG1 – DIG4, PTA12 – PTA31 ... tlačítka spínající zem.



PTA0-PTA7 ... segmenty A-F a desetinná tečka DP (katody)

PTA8-PTA11 ... anody jednotlivých číslicovek DIG1-DIG4

PTA12-PTA31 ... tlačítka spínající zem

Určete obsah registrů GPIOA_PDD, PORTA_PCR0, PORTA_PCR12 a zapište je v šestnáctkové soustavě do připravených rámečků. Bity 16-31 registrů PCR ponechejte nulové. Drive Strength postačí "Low", PFE vypněte, Slew Rate nastavte na Fast.

Řešení:

GPIOA_PDDR	0x0000 0FFF
PORTA_PCR0	0x0000 0100 (dioda PTA0)
PORTA_PCR12	0x0000 0103 (tlačítko PTA12)

GPIOA_PDDR: // PDDR urcuje, co jsou vstupy a co výstupy

- PTA0 az PTA7 jsou katody => výstupy, proto jsou nastaveny na 1
- PTA8 az PTA11 jsou anody => výstupy, proto jsou nastaveny na 1
- PTA12 az PTA31 jsou tlačítka, tedy vstupy a proto 0

PORTA_PCR0: katoda (dioda)

- bity 16-31 jsou 0 podle zadání,
- je to výstup a tedy PE a PS nemají význam

PORTA_PCR12: tlačítko

- bity 16-31 jsou 0 podle zadání,
- tlačítko spina zem, tedy na něm potřebujeme konstantně udržovat logickou 1
- to proto, abychom sepnuti zaznamenali, to se vykona pomocou pull-up rezistoru, a proto PE = 1 a PS = 1

Spodní polovina registru PORTx_PCRn:

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									0				0			
W									MUX		DSE		PFE		SRE	
Reset	0	0	0	0	0	*	*	*	0	*	0	*	0	*	*	*

10–8 MUX	<p>Pin Mux Control</p> <p>Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved and may result in configuring the pin for a different pin muxing slot.</p> <p>The corresponding pin is configured in the following pin muxing slot as follows:</p> <ul style="list-style-type: none"> 000 Pin disabled (analog). 001 Alternative 1 (GPIO). 010 Alternative 2 (chip-specific). 011 Alternative 3 (chip-specific). 100 Alternative 4 (chip-specific). 101 Alternative 5 (chip-specific). 110 Alternative 6 (chip-specific). 111 Alternative 7 (chip-specific).
7 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
6 DSE	<p>Drive Strength Enable</p> <p>This bit is read only for pins that do not support a configurable drive strength.</p> <p>Drive strength configuration is valid in all digital pin muxing modes.</p> <ul style="list-style-type: none"> 0 Low drive strength is configured on the corresponding pin, if pin is configured as a digital output. 1 High drive strength is configured on the corresponding pin, if pin is configured as a digital output.

Field	Description
5 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
4 PFE	<p>Passive Filter Enable</p> <p>This bit is read only for pins that do not support a configurable passive input filter.</p> <p>Passive filter configuration is valid in all digital pin muxing modes.</p> <ul style="list-style-type: none"> 0 Passive input filter is disabled on the corresponding pin. 1 Passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input. Refer to the device data sheet for filter characteristics.
3 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
2 SRE	<p>Slew Rate Enable</p> <p>This bit is read only for pins that do not support a configurable slew rate.</p> <p>Slew rate configuration is valid in all digital pin muxing modes.</p> <ul style="list-style-type: none"> 0 Fast slew rate is configured on the corresponding pin, if the pin is configured as a digital output. 1 Slow slew rate is configured on the corresponding pin, if the pin is configured as a digital output.
1 PE	<p>Pull Enable</p> <p>This bit is read only for pins that do not support a configurable pull resistor. Refer to the Chapter of Signal Multiplexing and Signal Descriptions for the pins that support a configurable pull resistor.</p> <p>Pull configuration is valid in all digital pin muxing modes.</p> <ul style="list-style-type: none"> 0 Internal pullup or pulldown resistor is not enabled on the corresponding pin. 1 Internal pullup or pulldown resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0 PS	<p>Pull Select</p> <p>This bit is read only for pins that do not support a configurable pull resistor direction.</p> <p>Pull configuration is valid in all digital pin muxing modes.</p> <ul style="list-style-type: none"> 0 Internal pulldown resistor is enabled on the corresponding pin, if the corresponding Port Pull Enable field is set.

Q:

Předpokládejte, že paměťové buňky na nejnižších adresách obsahují informaci dle této ilustrace:

Oblast paměti s částí tabulky vektorů přerušení				
Adresa	Číslo vektoru	Číslo IRQ	Obsah adresy	Význam
0x000000BC	47	31	0x200053C0	Port B
0x000000B8	46	30	0x20000100	Port A
...
0x00000070	28	12	0x20004E80	UART0
0x0000006C	27	11	0x20000100	-
0x00000068	26	10	0x20004CF0	SPI0
...
0x00000008	2	-	0x20003AB0	NMI
0x00000004	1	-	0x20002000	PC Init
0x00000000	0	-	0xFFFFFFFF	SP Init

Odpovězte na následující otázky (v šestnáctkové soustavě):

- Na které adrese začíná obsluha přerušení vyvolaného zdrojem UART0?
- Kterou hodnotu je nutno zapsat do registru ICPR, aby došlo k nulování případných nastavených příznaků přerušení vyvolaného zdrojem UART0?
- Kterou hodnotu je nutno zapsat do registru ISER, aby došlo k povolení přerušení vyvolaného zdrojem UART0?
- Obsluha kterého přerušení bude zahájena jako první, víte-li, že současně vzniknou požadavky na přerušení do zdrojů UART0, SPI0, Port A, Port B a žádné další požadavky neexistují ani nevzniknou? (odpovězte identifikací zdroje)

1. 0x20004EB0

// Tabulka vektorů přerušení: Obsah adresy je adresa, na kterou se má skočit pro // obsluhu přerušení z daného zdroje.

2. 0x0000 1000 (jednička na bitu s indexem 12)

// ICPR: Každý bit v registru reprezentuje jeden zdroj přerušení.

// Index bitu je stejný jako číslo IRQ. Např. bit s indexem 10 (v registru ICPR) // patří k SPI0 (podle tabulky viz výše je jeho číslo IRQ 10).

3. 0x0000 1000

// ISER: Stejný princip jako v ICPR. (Write 1 enable interrupt)

4. SPI0

// U ARM je možné konfigurovat prioritu.

// Defaultně mají výjimky s číslem 4 a výše všechny shodně prioritu 0.

// Pak se obslouží dříve ta výjimka, která má nižší číslo IRQ

// Z vypsaných má nejnižší IRQ SPI0, tedy se obslouží nejdříve.

Q:

Popište stručně (v bodech), co přinese (jaké výhody), pokud výrobce mikrokontroléru zvolí Von Neumannovskou architekturu mikrokontroléru. (3b)

Řešení:

Von Neumannovská x Harvardská architektura

- Jediná operační paměť, jediný adresový prostor.
- Flexibilnější pro (měnící se) aplikace.
- Možnost samomodifikujícího se kódu.
- Sběrnice je úzké hrdlo (instrukce a operandy k nim musíme z paměti vybírat postupně).
- Instrukce a data mají každá „svoji“ paměť – dvojí adresový prostor!
- Umožňuje použít fyzicky rozdílné technologie paměti – např. pro program FLASH a pro data RAM.
- Umožňuje použít rozdílnou velikost buňky paměti – např. 8 bitů pro data a 18 bitů pro instrukci.
- Umožňuje v jednom taktu získat instrukci a zároveň operandy pro ni.

Q:

Na základě analýzy níže uvedeného zdrojového textu programu stanovte hodnotu ukazatele zásobníku (SP) ve vyznačeném místě běhu programu. Odpověď v šestnáctkové soustavě (hex) vepište do orámovaného odpovědního boxu. (4b)

```
1      INCLUDE 'derivative.inc'
2      XDEF _Startup
3      ABSENTRY _Startup
4
5      STACK_START equ $10AF
6      LOCAL_BYTES equ 1
7      ORG ROMStart
8
9      _Startup:
10     ldhx #STACK_START+1
11     txs
12     lda #0
13
14     psha
15     jsr sub
16     loop: bra loop
17
18     sub:
19     lda 3, SP
20     ais #-LOCAL_BYTES
21 ; <--- (SP == ?) ---
22     ais #LOCAL_BYTES
23     rts
24
25     ORG $FFFFE
26     DC.W _Startup
```

// assembly prej nebudou, ale klidne nekdo muze odpovedet

Q:

Předpokládejte, že po dokončení instrukce cli je za stavu (tj. obsahu paměťových buněk a CPU registrů) daného níže uvedenou tabulkou vyvoláno povolené přerušení ze strany KBI. Určete obsahy registrů PC, SP bezprostředně před prováděním první instrukce příslušné obsluhy přerušení. Požadované odpovědi vepište v šestnáctkové číselné soustavě (hex) do orámovaných odpovědných boxů. (3b)

Address (High/Low)	Vector	Vector Name
0xFFCA:FFCB	ADC Conversion	Vadc
0xFFCC:FFCD	KBI	Vkeyboard
0xFFFFA:FFF8	IRQ	Virq
0xFFFFC:FFF9	SWI	Vswi
0xFFFFE:FFFF	Reset	Vreset

Paměťová buňka	CPU registr
Adresa (hex)	Obsah (hex)
FFCB	64
FFCC	1D
FFCD	0C
FFCE	19
FFCF	D3
Název	Obsah (hex)
A	1
H	0
X	AE
PC	1C48
SP	10A5

PC = 0x1D0C

// obsah adresy FFCC:FFCD je 1D:0C, na tuhle hodnotu se PC nastavi

SP = 0x10A0 // pro descending stack nebo 0x10AA pro ascending

// před vykonáním obsluhy prerušení se nejprve uloží obsah všech registrů na zasobník

// A,H,X se vejdu do 1 bajtu, PC se vejde do 2 => 1+1+1+2 = 5 bajtu

// tedy je potřeba posunout SP o 5 bajtu

// ARM využívá full-descending stack, cize 0x10A0 se jeví jako správnější výsledek

Q:

Slovy stručně uveďte ve správném pořadí, co musí programátor provést k realizaci příjmu jednoho bytu dat do proměnné TEMP na straně MCU pomocí rozhraní SCI.

Předpokládejte, že modul SCI byl již aktivován nastavením příslušného bitu v řídícím registru. Periferie, s níž je komunikováno, je také aktivní, podsystém přerušení v tomto případě není využit. (2b)

Zkontrolovat nastavení příslušného bitu a následně uložit hodnotu z registru do proměnné TEMP.

// TODO - not sure

Q:

Co je jádrem modulu SPI, umožňujícím sériový přenos dat? Napište název základní HW komponenty (obecně na úrovni číslicových obvodů), pomocí které je po jednotlivých bitech sériový přenos realizován. (1b)

posuvný registr, generátor časového signálu

Q:

Uveďte stručně, k čemu slouží tzv. "pull-up" rezistor na číslicovém vstupu? (2b)

Takzvaný pull up rezistor je v elektronice prvek činného odporu v logickém obvodu, za jehož pomoci se na vodiči nesoucí informaci určuje logická hodnota v době, kdy není definována žádným zařízením. To nastává v případě nezapojeného vstupu, vstupu, k němuž je připojena vysoká impedance, nebo výstupu s otevřeným kolektorem.

Q:

Uveďte příklad obvodu (stačí název, netřeba kreslit), připojeného na vstup mikrokontroléru, pro který pull-up rezistor vypnete. (2b)

Možností je veľa, ľubovoľný obvod, ktorý stále poskytuje logickú jednotku alebo nulu (t.j. nikdy nie je odpojený), najjednoduchší je napr. klopný obvod.

Z poznámok ku polsemke 2016:

Tato otázka má velmi mnoho správných odpovědí, obecně by to mohl být každý obvod, který na svém výstupu má v každém okamžiku nějakou platnou logickou úroveň (ty jsou jen dvě). Není pak třeba definovat klidovou úroveň pull-up rezistorem. Běžné hradlo například, klopný obvod, i ledacos složitějšího. Akceptoval jsem i třeba A/D převodník (a tiše doufal, že myslíte nějaký externí, ne ten uvnitř). Samotný jediný tranzistor to právě nebude - je bud' otevřený (a nějaké napětí se na vstup dostane) nebo zavřený - a právě v tom případě je třeba pull-up rezistor (nebo druhý, komplementární tranzistor).

Q:

Co je Drive Strength, na co je dobrý a na co se používá? (2b)

Drive Strength (pri GPIO se casto uvadi v mA) urcuje, kolik proudu nejake zarizeni vysle pri nejake zatezi (zatezi se myсли spotreba energie).

Neboli take urci, kolik zateze muze byt vedeno nejakou danou rychlosti bez poruseni integrity posilaneho signalu.

- prilis vysoky drive strength muze zpusobit, ze CPU bude ridit prilis vysoky proud, coz jej muze spalit
- prilis nizky drive strength muze zpusobit, ze se spali samotne GPIO
- nizke hodnoty znizuje prekmity

V podstate to jen specifikuje, ze pri tetto urovni proudu bude napeti stabilni.

Q:

Uveďte stručně, proč je u mikrokontrolérů řady HCS08 možno nastavit rychlosť změny úrovně (tzv. "Slew Rate") u výstupního pinu nějakého portu x pomocí registru PTxSE, tj. k čemu toto nastavení slouží, proč je potřebujeme. (2b)

Volba „slew rate“ zlepšuje parametry EMC – méně rušení. Sama strmá hrana zvyšuje rušení. Pokud je velký překmit, zafunguje ochranná dioda a tak vyvolaný proud způsobí opět rušení. To se stává zejména u dlouhých vodičů či při malých blokovacích kapacitách.

Sekce D: Pulsemka 2015 - skupina A

Q:

Definice vestavěného systému?

(3 správné odpovědi)

- Kombinace hardwaru a softwaru, jejímž smyslem je řídit externí proces, zařízení nebo systém.
- Počítač, který je zabudován do systému, ale pro uživatele není jako počítač viditelný.
- Slouží pro řízení celého systému a poskytuje výpočetní podporu pro jednotlivé komponenty systému.

Q:

Porovnání RAM, Flash paměti z hlediska kapacity, rychlosti zápisu a schopnosti uchovat data.

- flash paměť mává větší kapacitu než RAM
 - RAM je rychlejší než flash paměť
 - RAM je volatilní (bez proudu se ztratí data v ni), flash je nevolatilní
 - RAM spotrebúva menej energie (použitie v low-power aplikáciach)
- Nevolatilní = informace se uchovává i bez příslušného elektrického proudu (např. CD)

Q:

Co udělat, aby šlo vyvolat přerušení od nějakého modulu?

Nastavit hodnotu ISER [Interrupt Set-Enable Register] tak, aby bit reprezentující dany modul (cislo pinu, na který je modul pripojen) byl nastaven na 1.

Q:

Zabalte bajt 10100111 do datového rámce se sudou paritou, klidová hodnota je 1.

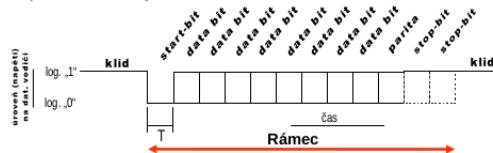
```
start bit      = 0      // start bit má opačnou hodnotu jako je klidový stav  
samotná data  = 10100111  
parity bit     = 1  
stop bits      = 11 // stop bit má vždy hodnotu klidového stavu  
// pouzijeme 2 stop bity, protože je nemame definovane (muze se pouzit i 1 stop bit)  
// paritni bit sude parity je v 1 proto, aby spolu s datami zabezpecil sudy pocet 1  
  
// pokud jde o asynchronní přenos, tak je LSB první  
time ->  
...011100101111...  
-----  
...sDDDDDDDDPSS... // s = start bit, S = Stop bit, P = parity bit, D = data
```

Datový rámec (frame) UART

První **informační (datový) bit** se na datovém vodiči objevuje až po nějaké době od počáteční hrany.

Tato doba je pochopitelně přesně dáná, zpravidla to bývá jeden bitový interval – jeden takt generátoru hodin vysílače (a jeden takt generátoru hodin příjmače, který je nyní s generátorem hodin vysílače zcela čerstvě zasynchronizován).

Tento bitový interval, který je vyplněn hodnotou opačnou, nežli je klidová hodnota a má význam hlavně kvůli hraničním, kterou na svém začátku způsobí, se nazývá **start bit**.



Stop bity

Za posledním **vyslaným datovým bitem** a případným paritním bitem se vždy musí vyslat alespoň jeden tzv. **stop bit**, který má vždy hodnotu klidového stavu (zde log. 1). Stop bit slouží k vzájemnému oddělení přenášených slov. Lze zpravidla nastavit počet stop bitů v rozsahu jeden, či dva stop bity.

Stop bity, kromě toho, že zajistí na datovém vodiči klidový stav, z něhož lze hranou přejít do start bitu následujícího slova, **slouží také k tomu, aby měl přijímač čas odeslat právě přijaté slovo ze vstupního posuvného registru dále ke zpracování**.

Trvání přenosu rámce

Necht' **n** je celkový počet bitů rámce a **T** je perioda synchronizačního signálu. Pak doba, za kterou je rámec přenesen, je $tr = n * T$.

n je vyjádřeno ve tvaru:

$$n = 1 + D + P + S,$$

kde „1“ na začátku vyjadřuje jediný start bit,

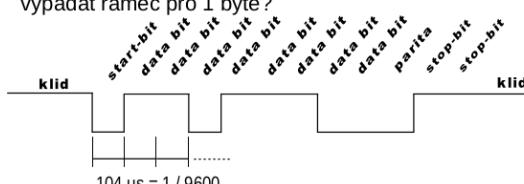
D je počet datových bitů,

P je počet paritních bitů (obvykle 0 nebo 1),

S je počet stop bitů.

Příklad asynchronního přenosu dat

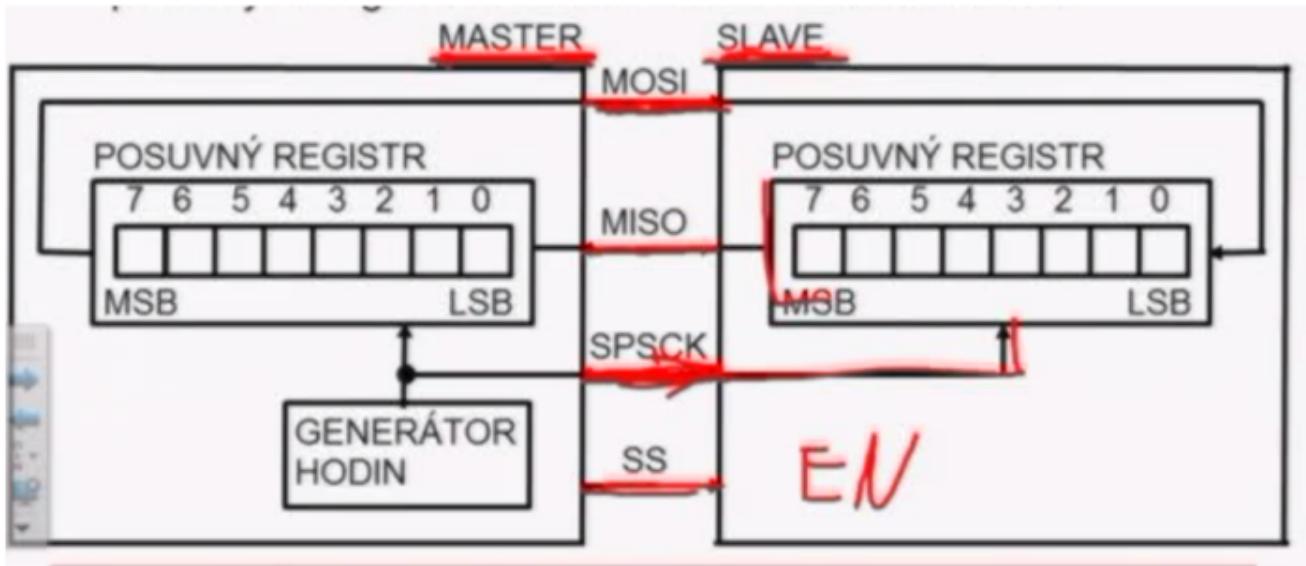
Necht' přenášený bajt má hodnotu 0x3B, tedy binárně 00111011. Přenos bude doplněn lichou paritou a dvěma stop bity. Jak bude vypadat rámec pro 1 byte?



Přenosová rychlosť nechť má být 9600 bd a je třeba přenést 1024 bytů dat. Jak dlouho přenos potrvá?

$$TP = 1024 * 12 / 9600 = 1,28 \text{ s}$$

Q:
blokové schéma SPI, popsat



Datové vodiče se na mikrokontroléru označují jako MISO a MOSI. Je tím vyjádřen směr toku dat podle režimu, v němž se dané zařízení připojené na SPI rozhraní nachází.

MISO = Master In, Slave Out = master je input, slave je output
MOSI = naopak

Pokud může zařízení pracovat pouze v režimu slave, což bývá časté u periferií s rozhraním SPI, jsou datové vodiče označeny jednoduše jako SO (výstup) a SI (vstup).

SPSCK (SPI Serial Clock) - generuje hodinový signál, na master je to vždy výstup a na zařízení slave vstup => hodinový signál vysílá právě zařízení master.

Výběrový signál SS (Slave Select), který je aktivní v log. 0, slouží k výběru a aktivaci zařízení, se kterým má probíhat komunikace.

Na zařízení master je signál SS výstupem připojeným k modulu slave. Ten je možné prostřednictvím mastera signálem SS aktivovat a zahájit tak datový přenos.

Master musí zajistit, že SS bude po dobu přenosu v log. 0 (aktivní). V daném čase smí probíhat komunikace pouze s jediným modulem slave, ostatní zařízení, jejichž SS vstup je v log. 1 (neaktivní) do komunikace nezasahují.

Na zařízení slave je signál SS vstupem, který v aktivní úrovni způsobí synchronní odesílání obsahu posuvného registru na stranu mastera.

Sekce E: Pulsemska 2019 - skupina A

Q:

1. LED je katodou pripojena na GND a anodou na pin mikrokontroleru. Frekvencia PWD je 1kHz a log.1 je na pine 400us. Vyjadri intenzitu rozsvietenia LED v %.

[MCU]=pin=--->|---GND => aby svitila, tak na pinu musi byt logicka jednicka

f = 1Khz = 1000 Hz => 1/1000 sec = 1 ms = 1000 us = T -> logicka 1 je na pinu 400 us, po zbytek casu je tam logicka 0... 400 us / 1000 us = 40%

// kdyby byla katoda spojena s pinem a anoda s Vcc, tak by nas zajimalo,
// jak casto je na pinu logicka nula, protoze chceme rozdil napeti, aby dioda svitila

Q:

2. UART ma 19200Bd. Datove slovo ma 8b + 2 stop_bit. Vypočítaj maximalnu dobu po ktoru musí byť prijaté datové slovo spracované.

Přenosová rychlosť se nejčastěji udává v **baudech (Bd)** což v případě binárního kódování vyjadřuje „bitů za sekundu“.

Protože u popsaného přenosu je pouze dvoustavová modulace, v každé periodě **T** lze přenést pouze informaci o velikosti 1 bit – buďto stav log. 0 nebo log. 1. Stav datového vodiče se uprostřed tohoto intervalu nesmí změnit, změny probíhají jen na hranicích. Z toho vychází, že **1 Bd = 1 bit/sec** a přenosová rychlosť tak odpovídá frekvenci hodinového signálu.

Při určité přenosové rychlosti bude však skutečný počet přenášených **datových bitů** (tzv. **efektivní přenosová rychlosť**) nižší, což je to dáné tím, že rámec obsahuje několik **režijních bitů** (start bit, případný paritní bit a stop bity), které nenesou žádná data.

Je-li $n = 1+8+\text{parita}+\text{stop_bity}$ je počet bitů rámce a N počet bytů přenášených dat, pak doba přenosu přenosovou rychlosť BAUD je

$$\text{TP} = N * n / \text{BAUD} [\text{s}]$$

1000000 microsec ... 19200 bits
x microsec ... 12 bits (1 + 8 + 1 + 2)
// 1 byte prenasanych dat, 12 bitov pre kazdy ramec
x/1000000 = (12 * 1)/19200 => x = (12*1000000) / 19200 = 625 microsekund (i guess?)

Q:

3. Bol obrazok PORT_PCR registru s dokumentacie. Na daný port je napojene Vcc cez tlacitko. Ked sa stlaci tlacitko tak sa ma vyvolat prerusenie. Urcit hexa obsah registru PORT_PCR.

K zodpovezeni tehole otazky je potreba mit dokumentaci hornich 16 bitu registru PORTx_PCR. Typicky je ale potreba nastavit ISF na 1 a IRQC na 1001.

```
// chceme pouzit rising edge (1001), tedy v momente zepnuti at se noco stane...
// jelikoz mame Vcc i tlacitko napojeno na stejny port, tedy se bude jednat o
// pulldown rezistor => stlaceni tlacitka tedy zmeni signal z 0 na 1,
// co je ten rising edge, ktery chceme detekovat
// nemuzeme pouzit 1100 (when logic 1), to by generovalo interrupt po celou dobu,
// kdy bude tlacitko sepnute, co nechceme
```

```
PORTx_PCR = 0x01090000 // s tim IRQC si ale nejsem 100% jisty
```

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R								ISF								
W								w1c								IRQC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTx_PCRn field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 ISF	Interrupt Status Flag This bit is read only for pins that do not support interrupt generation. The pin interrupt configuration is valid in all digital pin muxing modes.
	0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic one is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 IRQC	Interrupt Configuration This field is read only for pins that do not support interrupt generation. The pin interrupt configuration is valid in all digital pin muxing modes. The corresponding pin is configured to generate interrupt/DMA request as follows: 0000 Interrupt/DMA request disabled. 0001 DMA request on rising edge. 0010 DMA request on falling edge. 0011 DMA request on either edge. 1000 Interrupt when logic zero. 1001 Interrupt on rising edge. 1010 Interrupt on falling edge. 1011 Interrupt on either edge. 1100 Interrupt when logic one. Others Reserved.

Q:

4. Modul RTC pocita sekundy. Aka ma byt hodnota 32b registru(binarnie), s ktorym sa RTC porovnava (pri zhode dojde k preruseniu), aby k preruseniu doslo presne za 2 dni.

Co umí modul RTC

- Počítat sekundy 32bitovým čítačem! 2^{32} sekund je více než 136 let!
- Může vyvolávat přerušení,
- může vzbudit MCU z režimu spánku
 - každou sekundu,
 - po uplynutí určitého počtu sekund (nastavení „budíku“).
- Sekundové pulsy lze vyvést na pin.
- Může být nezávislý na zdroji hodin pro zbytek MCU.
- Resetuje se jen při zapnutí napájení.
- Potřebuje k tomu jen velmi málo energie.

*čas měřený modulem RTC je v rádech, které jsou smysluplné v reálném světě - žádné mikrosekundy, ale sekundy.

2 dni = 2 * 24 hod * 60 min * 60 sekund = 172800 sekund

172800(dec) = 2A300(hex)

obsah registru RTC bude v casu preruseni 0x0002A300 => s touhle hodnotou se bude porovnovat obsah RTC pro potvrzeni vzniku preruseni
jelikoz to chcou binarne, tak: 0b0000_0000_0000_0010_1010_0011_0000_0000

Sekce F: Pulsemka 2021 - skupina A

Q:

Popsat "Bit-binding", co to je a na co se to využívá?

Bit-band operace umožňuje jednou load/store operaci přistoupit ke konkrétnímu bitu paměti, neboli umožňuje změnu jednoho bitu v 32-bitovém kusu dat.

Místo bezného postupu (nacít 32bitovou hodnotu dat, změnit jeden bit přes masku, zapsat změněnou hodnotu zpátky) je možno přes "bit-band alias address" primo zapsat konkrétní bit.

Bit-band Operations

- Bit-band operation allows a single load/store operation to access a single bit in the memory, for example, to change a single bit of one 32-bit data:

- Normal operation without bit-band (read-modify-write)
 - Read the value of 32-bit data
 - Modify a single bit of the 32-bit value (keep other bits unchanged)
 - Write the value back to the address
- Bit-band operation
 - Directly write a single bit (0 or 1) to the "bit-band alias address" of the data

Risky in some cases!

- Bit-band alias address

- Each bit-band alias address is mapped to a real data address
- When writing to the bit-band alias address, only a single bit of the data will be changed

Q:

ReceiveChar() // viz [Sekce B: Pulsemka 2017 - skupina A](#)

Q:

Nakonfigurovať tlačidlo, aby sa pri stlacení vyvolalo prerušenie (IRQC, MUX, PS, PE).

// IRQC = 1001 (rising edge, možno použiť i log. 1), MUX = 001 (GPIO),

// PE = , PS = 0 (pulldown), PE = 1

PORTx_PCRn = 0x00090102; (x,n podľa zadania)

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R				0					ISF			0				
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			0						MUX		0	DSE	0	PFE	0	SRE
W										0	*	*	*	0	*	PS
Reset	0	0	0	0	0	*	*	*	0	*	0	*	0	*	*	*

PORTx_PCRn field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 ISF	Interrupt Status Flag This bit is read only for pins that do not support interrupt generation. The pin interrupt configuration is valid in all digital pin muxing modes.

	<p>0 Configured interrupt is not detected.</p> <p>1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic one is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.</p>																				
23–20 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>																				
19–16 IRQC	<p>Interrupt Configuration</p> <p>This field is read only for pins that do not support interrupt generation.</p> <p>The pin interrupt configuration is valid in all digital pin muxing modes. The corresponding pin is configured to generate interrupt/DMA request as follows:</p> <table> <tr><td>0000</td><td>Interrupt/DMA request disabled.</td></tr> <tr><td>0001</td><td>DMA request on rising edge.</td></tr> <tr><td>0010</td><td>DMA request on falling edge.</td></tr> <tr><td>0011</td><td>DMA request on either edge.</td></tr> <tr><td>1000</td><td>Interrupt when logic zero.</td></tr> <tr><td>1001</td><td>Interrupt on rising edge.</td></tr> <tr><td>1010</td><td>Interrupt on falling edge.</td></tr> <tr><td>1011</td><td>Interrupt on either edge.</td></tr> <tr><td>1100</td><td>Interrupt when logic one.</td></tr> <tr><td>Others</td><td>Reserved.</td></tr> </table>	0000	Interrupt/DMA request disabled.	0001	DMA request on rising edge.	0010	DMA request on falling edge.	0011	DMA request on either edge.	1000	Interrupt when logic zero.	1001	Interrupt on rising edge.	1010	Interrupt on falling edge.	1011	Interrupt on either edge.	1100	Interrupt when logic one.	Others	Reserved.
0000	Interrupt/DMA request disabled.																				
0001	DMA request on rising edge.																				
0010	DMA request on falling edge.																				
0011	DMA request on either edge.																				
1000	Interrupt when logic zero.																				
1001	Interrupt on rising edge.																				
1010	Interrupt on falling edge.																				
1011	Interrupt on either edge.																				
1100	Interrupt when logic one.																				
Others	Reserved.																				

10–8 MUX	<p>Pin Mux Control</p> <p>Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved and may result in configuring the pin for a different pin muxing slot.</p> <p>The corresponding pin is configured in the following pin muxing slot as follows:</p> <table> <tr><td>000</td><td>Pin disabled (analog).</td></tr> <tr><td>001</td><td>Alternative 1 (GPIO).</td></tr> <tr><td>010</td><td>Alternative 2 (chip-specific).</td></tr> <tr><td>011</td><td>Alternative 3 (chip-specific).</td></tr> <tr><td>100</td><td>Alternative 4 (chip-specific).</td></tr> <tr><td>101</td><td>Alternative 5 (chip-specific).</td></tr> <tr><td>110</td><td>Alternative 6 (chip-specific).</td></tr> <tr><td>111</td><td>Alternative 7 (chip-specific).</td></tr> </table>	000	Pin disabled (analog).	001	Alternative 1 (GPIO).	010	Alternative 2 (chip-specific).	011	Alternative 3 (chip-specific).	100	Alternative 4 (chip-specific).	101	Alternative 5 (chip-specific).	110	Alternative 6 (chip-specific).	111	Alternative 7 (chip-specific).
000	Pin disabled (analog).																
001	Alternative 1 (GPIO).																
010	Alternative 2 (chip-specific).																
011	Alternative 3 (chip-specific).																
100	Alternative 4 (chip-specific).																
101	Alternative 5 (chip-specific).																
110	Alternative 6 (chip-specific).																
111	Alternative 7 (chip-specific).																

1 PE	<p>Pull Enable</p> <p>This bit is read only for pins that do not support a configurable pull resistor. Refer to the Chapter of Signal Multiplexing and Signal Descriptions for the pins that support a configurable pull resistor.</p> <p>Pull configuration is valid in all digital pin muxing modes.</p> <ul style="list-style-type: none"> 0 Internal pullup or pulldown resistor is not enabled on the corresponding pin. 1 Internal pullup or pulldown resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0 PS	<p>Pull Select</p> <p>This bit is read only for pins that do not support a configurable pull resistor direction.</p> <p>Pull configuration is valid in all digital pin muxing modes.</p> <ul style="list-style-type: none"> 0 Internal pulldown resistor is enabled on the corresponding pin, if the corresponding Port Pull Enable field is set. 1 Internal pullup resistor is enabled on the corresponding pin, if the corresponding Port Pull Enable field is set.

Q:

Máme nejaký motor na porte A, ktorý kontroluje pohyby zariadenia v 3 osiach, každá os zodpovedá nejakému pinu (piny 10 až 12 pre osy x až z). Bolo vyvolané prerušenie, ktorého obsluha vyžaduje zastaviť pohyb motora na osi x. Napište prikaz v C, ktorý zastavi pohyb motora v osi x. // pin 10 = os x

```
// os x = pin 10
GPIOA->PCOR = (1 << 10); // cez PCOR se jednoduseji riadi PDOR, neni treba maskovat
```

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

GPIOx_PDOR field descriptions

Field	Description
31-0 PDO	<p>Port Data Output</p> <p>Register bits for un-bonded pins return a undefined value when read.</p> <p>0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output.</p> <p>1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output.</p>

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

GPIOx_PCOR field descriptions

Field	Description
31-0 PTCO	<p>Port Clear Output</p> <p>Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows:</p> <p>0 Corresponding bit in PDORn does not change.</p> <p>1 Corresponding bit in PDORn is cleared to logic 0.</p>

Sekce G: Pulsemka 2021 - skupina B

Q:

Popište, čím se vyznačuje Von Neumannova architektura z hlediska paměti. Také popište, jakou výhodu tahle architektura z hlediska paměti poskytuje v rámci embedded systems.

- Jediná operační paměť, jediný adresový prostor.
- Flexibilnejší pro (měnící se) aplikace.
- Možnost samomodifikujícího se kódu.

Q:

```
SendChar() // viz Sekce B: Pulsemka 2017 - skupina A
```

Q:

Napište příkaz v C, který vyvolá přerušení na pinu 4 portu A.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R					0			ISF		0						
W								w1c								IRQC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTx_PCRn field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 ISF	Interrupt Status Flag This bit is read only for pins that do not support interrupt generation. The pin interrupt configuration is valid in all digital pin muxing modes.
	0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic one is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

```
PORTA_PCR4 |= (1 << 24); // OR, protoze nechceme porusit konfiguraci registru
```

Q:

Mame nejaký motor na portu B, co se pohybuje po 3 osach, které jsou na pinech 10-12 (os x az z). Bylo vyvolano prerušení. V rámci obslužní rutiny napiste příkaz v C, který nastaví os y do logické nuly, aby se při zachycení prerušení prestal motor v tomto směru tocit.

```
GPIOB->PCOR = (1 << 11); // podobně, jak ve skupině A
```

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R																	PDO																
W																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIOx_PDOR field descriptions

Field	Description
31–0 PDO	Port Data Output Register bits for un-bonded pins return a undefined value when read. 0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output. 1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R																	0																
W																	PTCO																

GPIOx_PCOR field descriptions

Field	Description
31–0 PTCO	Port Clear Output Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is cleared to logic 0.