Cv3.

Kontrolní otázky:

**1. Proč je v programech u mikropočítačů obvykle použita nekonečná smyčka.**Předpokládá se, že poběží navždy

**2. Jaké výhody a nevýhody má použití nekonečné smyčky.**  
+ Jsou jednoduché,   
+ Jsou efektivní, mají malé požadavky na hardware, např. MCU nemusí mít modul časovače.  
+ Jsou snadno přenositelné – budou pracovat na jiném MCU s minimální potřebou změn v kódu.

- Nejsou přesné – pokud je potřeba měřit data každých 5 ms, tento typ systému neposkytne dostatečnou přesnost ani pružnost  
- V základní verzi pracuje systém stále „na plný výkon“. To znamená, že má velkou spotřebu energie.

**3. Jak se realizuje softwarové zpoždění? Jaké má výhody a nevýhody.**  
Jednoduché zpoždění, které nevyžaduje žádný speciální hardware (časovač):   
void delay(void) {  
long cnt;   
for ( cnt=0; cnt < 30000; cnt++);

}

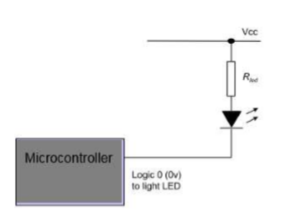
+ Mohou vytvořit velmi krátká zpoždění.  
+ Nevyžadují žádné hardwarové časovače.  
+ Budou fungovat na libovolném mikropočítači.

- Je obtížné vytvořit zpoždění s přesnou dobou trvání.  
- Smyčka (smyčky) musí být znovu „naladěny“ pokud se rozhodneme použít jiný mikropočítač, změníme nastavení hodinové frekvence procesoru nebo jen změníme nastavení optimalizací překladače.

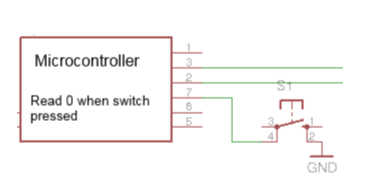
**4. Jak se liší diskrétní a analogové vstupy?**Diskretni – 0 a 1  
Analog – 0 až 1

**5. Uveďte příklady diskrétních vstupů a výstupů.**klasicke svetlo – zapnuto X vypnuto  
SmartSvetlo – off, minimum, medium, max

**6. Nakreslete schéma připojení LED k mikropočítači.**

****

**7. Nakreslete schéma připojení tlačítka k mikropočítači.**

****

**8. Co jsou to zákmity tlačítka? Jaké problémy mohou způsobit?**Zákmity znamenají, že kontakt se opakovaně sepne a rozepne během krátkého času  
- Místo „A“ přečte program z klávesnice „AAAA“.  
- Je obtížné počítat, kolikrát bylo tlačítko stisknuto.  
- Pokud je tlačítko stisknuto a po nějaké době uvolněno, kvůli zákmitům se to může v programu jevit, jakoby bylo stisknuto znovu (v okamžiku, kdy je ve skutečnosti uvolněno).

**9. Popište možnosti ošetření zákmitů tlačítek v programu.**Ošetřit zákmity lze buď hardwarově, vhodnou konstrukcí obvodu mezi tlačítkem a MCU. Nebo softwarově, např. takto:  
- Přečíst hodnotu na pinu.  
- Pokud se zdá, že bylo tlačítko stisknuto, počkat asi 20 ms a pak znovu přečíst pin.  
- Pokud druhé čtení potvrzuje výsledek prvního (tlačítko stále stisknuto), předpokládáme, že tlačítko bylo opravdu stisknuto.

**CV4)**

**Kontrolní otázky   
1. Vysvětlete důvody pro používání funkcí v programu, jaké jsou hlavní přínosy?**Čitelnější kód, pokud se něco opakuje, tak to nemusíme psát několikrát, zjednodušení, přehlednost

**2. Vysvětlete význam klíčového slova inline a static u funkce. Jaký je rozdíl mezi inline a „obyčejnou“ funkcí?**If a function is inline, the compiler places a copy of the code of that function at each point where the function is called at compile time. Normal functions do not have any such functionality. In case of inline function, function calling is replaced by that function definition.

**3. K čemu slouží registr GPIO PDDR (Port Data Direction Register)?**Nastavuje jednotlivé piny jako vstupy nebo výstupy.

**4. K čemu slouží registr GPIO PDOR (Port Data Output Register)?**Nastavuje logickou hodnotu (napěťovou úroveň) na pinech, které jsou nakonfigurovány jako výstupy.  
0 – na pinu bude nastavena log. 0 (napětí 0 V)   
1 – na pinu bude nastavena log. 1 (napětí 3 V)

**5. K čemu slouží registr GPIO PDIR (Port Data Input Register)?**   
Z tohoto registru se čte hodnota na pinu, který je nakonfigurován jako vstup. Viz 4

**6. Co je potřeba nastavit/povolit, abychom mohli pin využívat jako GPIO vstup nebo výstup?   
a) Povolit (zapnout) hodinový signál pro příslušný port**Příklad: povolení hodinového signálu pro port A a port B:   
SIM->SCGC5 |= (SIM\_SCGC5\_PORTA\_MASK | SIM\_SCGC5\_PORTB\_MASK );   
Masky jako SIM\_SCGC5\_PORTA\_MASK jsou definovány v hlavičkovém souboru MKL25Z4.H.

**b) Nastavit funkci příslušných pinů do režimu GPIO**Příklad: nastavení pinu do režimu GPIO:   
PORTB->PCR[19] = PORT\_PCR\_MUX(1);

**7. Napište kód, který nastaví pin 7 na portu A (tj. pin PTA7) na funkci GPIO. Využijte dokumentaci mikropočítače.**

**8. Jaké funkce může mít pin u mikropočítače? Uveďte několik příkladů.**   
Výstup dat pro sériovou komunikaci (UART Tx), vstup dat pro A/D převodník apod

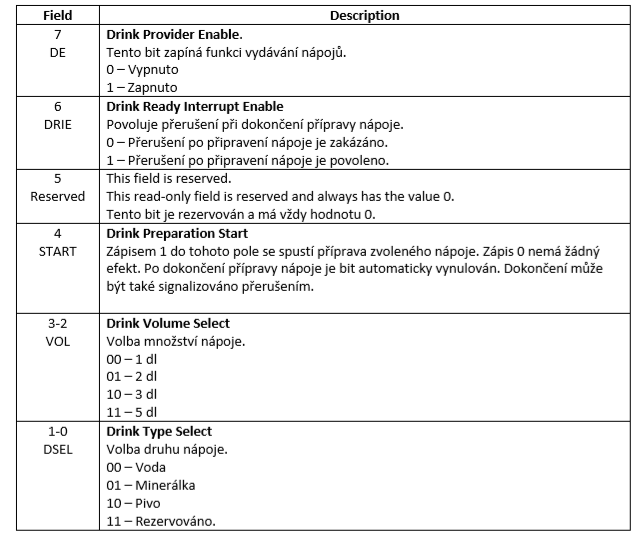
**9. Otázky podle testu „Test bitové operace v C“.**

**10. Otázky podle testu „Test komunikace MCU“.**

**CV5) bez otázek**

**CV6)**

**1. K čemu slouží registry periferií? Uveďte příklady.**Registry sloužící pro ovládání periferii, např. AD převodníku, časovače, atd. Periferií je zde myšlen obvod mimo procesor, ale uvnitř čipu (on-chip peripheral).

**2. Vysvětlete, jak se ovládá interní periferie mikropočítače, např. A/D převodník.  
**

**3. K čemu slouží analogově digitální převodník?**K převodu analogového signálu na digitální

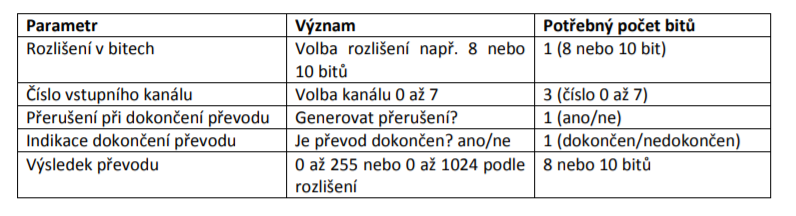
**4. Jaký analogový signál je na vstupu mikropočítače?**

**5. Co je vstupem A/D převodníku a co je jeho výstupem.**Diferenciální, označené DADPn a DADMn (až 4 kanály = 4 páry vstupů)   
Jednoduché (single-ended), označené ADn (až 24 vstupních kanálů)

**6. Jaké rozlišení má A/D převodník našeho mikropočítače KL25Z?**   
Rozlišení až 16 bitů

**7. Uveďte příklady vlastností A/D převodníku, které je možno nastavit.**

Rozlišení v bitech • Číslo vstupního kanálu • Povolení přerušení při dokončení převodu • Indikace dokončení převodu • Výsledek převodu



**8. Jestliže má A/D převodník rozlišení 10 bitů, v jakém rozsahu je jeho výstupní hodnota?**

2^0 – (2^n)-1, t.j. v našom prípade 0 – 1023 vrátane krajných hodnôt.

**9. Kde je uložen výsledek převodu A/D převodníku?**

Výsledok prevodu A/D prevodníku je zvyčajne uložený do tzv. dátového registru periférie (slúži na medzivýsledky, či jako buffer před odoslaním dát po linke).

**10. K čemu je u A/D převodníku využito přerušení?**

K čítaniu / zápisu nameranej hodnoty v pravidelných intervalech udávaných prerušením od časovača, príp. K tomu, aby sám A/D prevodník generoval prerušenie při dokončení prevodu a dal tým najavo, že hodnota je pripravená pre ďalšie spracovanie.

**11. Vysvětlete důvody pro ovladačů zařízení (periferií) v programu, jaké jsou hlavní výhody?**

Poskytujú dodatočnú vrstvu abstrakcie – programátor nemusí pracovať priamo s adresami, registrami a polohou jednotlivých bitov ale volá funkcie poskytované ovládačom. Hlavnou výhodou je teda nižšia chybovosť, zvýšenie časovej efektivity, čitateľnosti kódu a v neposlednej rade omnoho lepšia prenositeľnosť, keďže ovládač môže na „skrytej“ úrovni riešiť rozdiely medzi rôznymi zariadeniami, pričom programátorovi ponúka len interface fungujúci všade rovnako.

**CV7)**

1. **Popište časovač u mikropočítače časovače – jaké má hlavní části? Jak funguje.**Základní částí časovače je čítač (viz podrobné vysvětlení níže). To je vlastně proměnná, jejíž hodnota se zvětšuje nastavitelnou rychlostí.

Princip přesného čekání:

• Nastavit časovač tak, aby běžel požadovanou rychlostí. Např. aby se čítač zvětšil o 1 každou milisekundu • Vynulovat čítač   
• Čekat až hodnota čítače dosáhne požadované doby čekání, např. 100 (ms).

**2. Jak je možno pomocí časovače pozastavit program na určitou dobu?**využít přerušení od časovače. Modul časovače můžeme nastavit tak, aby generoval přerušení v určitém intervalu a v obsluze přerušení pak vykonat potřebnou akci.

Princip:   
• Nastavit časovač tak, aby generoval přerušení s požadovanou periodou, např. každých 100 ms. • Do obsluhy přerušení umístit kód, který má být pravidelně vykonáván s touto periodou.

**3. Jakým způsobem můžeme pomocí časovače zajistit periodické vykonání nějaké akce, např. přepnutí stavu LED každých 100 ms?**

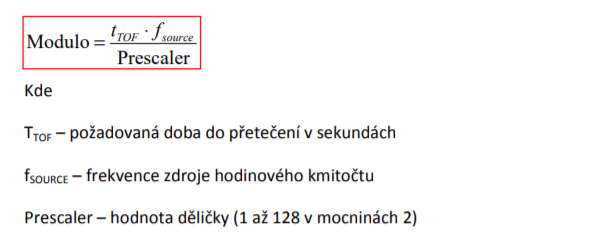
Zapísaním hodnoty do Modulo registru časovača tak, aby pretiekol každých 100 ms a vygeneroval prerušenie. Následne umiestnením kódu do obsluhy prerušenia, ktorá vykonáva potrebnú činnost. Podrobnejšie viď. Otázka 6.

**4. Kolik bitů má čítač časovače TPM?**16

**5. K čemu slouží u časovače TPM modulo registr?**   
Určí hodnotu, kdy registr přeteče

**6. Pokud je frekvence čítače 1 MHz a v modulo registru je hodnota 1000, po jaké době dojde k přetečení čítače (událost TOF)?**

Záleží ešte na Prescale – rozlíšenie deličky (štandard 128). Následne vzorec:



a teda Ttof = (Modulo \* Prescaler) / fsource

Ttof = (1000 \* 128) / 1000000 = 0.128s

**7. K čemu lze u mikropočítačů využít přerušení?**Přerušení umožňuje přinutit procesor reagovat na vnější podnět. Pomocí přerušení je možno procesoru signalizovat různé události, např. že

• uživatel stiskl tlačítko   
• uplynulo určité množství času   
• přišla nová data ze sériového rozhraní   
• A/D převodník dokončil převod

**8. Jakou zkratkou se v angličtině označuje obslužná rutina přerušení? Co tato zkratka znamená?**ISR – Interrupt Service Routine. ISR je funkce, která je volána, když nastane dané přerušení. Úkolem této funkce je zpracovat přerušení a pak předat řízení zpět do hlavního programu.

**9. Vysvětlete co je to souběh (race condition).**

Je to situácia, kedy viacero vlákien či procesov súčasne pristupuje ku zdieľaným zdrojom. Môže dôjsť k viacerým chybám vyplývajúcim z toho, že čítanie a zápis z pohľadu procesoru nie sú atomickými operáciami a môžu byť kedykoľvek prerušené (napr. Počas prepisovania hodnoty, ktorú v inom vlákne práve čítame).

**10. Co znamená zkratka PWM?**PWM je zkratka Pulse-Width modulation. Doslova tedy modulace šířky pulsu.   
Příklady použití PWM:  **11. K čemu lze využít PWM?**• Přenos informace   
• Řízení výkonu elektrických strojů, např. topné těleso, motor   
• Jednoduchá náhrada Digitálně-Analogového převodníku (DAC)  
  
**CV8) bez otázek**

**CV9) bez otázek**

**CV10)**

**1. Co znamená zkratka RTOS?**   
RTOS je zkratka pro operační systém reálného času (Real Time Operating Systém- RTOS).

**2. Jak se liší RTOS od běžného operačního systému?**Musí pracovat v real time, takže musí vždy vykonat do předem dané doby

**3. Co je to priorita procesu? Jak se vyjadřuje?**Číslo ozančující, jak moc důležitá operace je a v jakém pořadí operace budou vhozeny na procesor

**4. Proces A má prioritu 1 a proces B má prioritu 2. Který proces má vyšší prioritu, tj. dostane se přednostně na procesor?**B

**5. Jak se jmenuje funkce FreeRTOS pro vytvoření procesu (úlohy)?**xTaskCreate()

**6. Co je to atomická operace?**Atomická operace Atomická operace = nedělitelná; z pohledu ostatních procesů se jeví vždy jako jedna operace, která už se neskládá z žádných dílčích úkonů.

**7. Vysvětlete, jak vzniká souběh a jak se mu dá zabránit pomocí semaforu (mutexu).**Přepínáním procesů, Vzájemným vyloučenímPrincip vzájemného vyloučení: (mutex)

• Synchronizační objekt může získat jen 1 proces (požádá o něj OS a ten mu jej přidělí)   
• Dokud proces objekt „nevrátí“, nemůže jej získat jiný proces   
• Každý proces před prací se sdíleným prostředkem musí získat synchronizační objekt.

**Bitové operace**:  
https://www.rapidtables.com/convert/number/binary-to-hex.html

**1. Zapište kód v C, který nastaví na 1 bit 1 v registru PTCD.**

PTCD |= 0b00000010 nebo PTCD |= 0x02 (celé: PTCD = PTCD | 0b00000010)

**2. Zapište kód v C, který nastaví na 1 bity 6 a 7 v registru PTCD.**

PTCD |= 0b11000000 nebo PTCD |= 0xC0

**3. Zapište kód v C, který vynuluje bit 0 v registru PTCD.**

PTCD &= 0b11111110 nebo PTCD &= 0xFE

**4. Zapište kód v C, který vynuluje bity 1 a 2 v registru PTCD.**

PTCD &= 0b11111001 nebo PTCD &= 0xF9

**5. Zapište kód v C, který posune (shift) bity v proměnné “mask” o 2 bity doleva.**

mask <<= 2 (celé: mask = mask << 2)

**6. Jestliže u předchozí úlohy obsahovala proměnná hodnotu 0x18 před posunem, jaká bude její**

**hodnota po posunu?**

původně: 00011000

teď: 01100000

**7. Zapište v C podmínku, která bude splněna, pokud bude bit 0 v registru PTAD mít hodnotu 1.**

if(PTCD & 0b00000001)