Odpovědi pište na zvláštní odpovědní list s vaším jménem a fotografií. Pokud budete odevzdávat více než jeden list s řešením, tak se na 2. a další listy nezapomeňte podepsat a do jejich záhlaví napsat i/N (kde i je číslo listu, N je celkový počet odevzdaných listů).

Společná část pro otázky označené X

Předpokládejte, že máme osobní počítač s následující architekturou:

① Hlavní systémová sběrnice je paralelní 66 MHz sběrnice s 36-bit adresovou a dedikovanou 64-bit datovou částí. Na systémovou sběrnici jsou připojeny 2 CPU Intel Pentium Pro (32-bit procesor s instrukční sadou x86, s 36-bit paměťovým adresovým prostorem, 16-bit I/O adresovým prostorem, 64-bit datovou sběrnicí; taktovací frekvence jádra je 166 MHz, součástí procesoru je 512 kB L2 cache, a 32 kB L1 cache, obě cache mají velikost řádky 32 B).

② Třetím zařízením připojeným na systémovou sběrnici jsou čipy Intel 82441FX „PCI and Memory Controller“ (PMC) a Intel 82442FX „Data Bus Accelerator“ (DBX), které dohromady tvoří northbridge chipsetu Intel 440FX. Northbridge je připojen na všechny datové vodiče systémové sběrnice, ale jen na adresové vodiče HA3 až HA31 (počítáno od 0). Northbridge v sobě obsahuje řadič paměti pro SIMM paměťové moduly DRAM – paměťová sběrnice je paralelní s 64-bit datovou a dedikovanou 30-bit adresovou částí). Na základní desce je v SIMM modulech nainstalován plný 1 GB maximální northbridgem podporované kapacity DRAM. Součástí northbridge je též PCI host bridge pro paralelní 33 MHz 32-bit sběrnici PCI (sdílená 32-bit adresová a datová sběrnice). Sběrnice PCI má oddělený paměťový a I/O adresový prostor.

③ Na sběrnici PCI je připojen southbridge Intel 82371SB „PCI I/O IDE Xcelerator“ (PIIX3). Stejná PCI sběrnice je též vyvedena do 4 PCI slotů na základní desce. PIIX3 v sobě mimo jiné obsahuje PCI-ISA bridge pro 16-bit ISA sběrnici (paralelní 8 MHz sběrnice, 16-bit datová a dedikovaná 24-bit adresová sběrnice, zvláštní 16-bit I/O adresový prostor). Tato ISA sběrnice je vyvedena do 2 ISA slotů na základní desce.

④ Ve 2. PCI slotu je vložena grafická karta S3 Trio V64. Ve 3. PCI slotu je vložen USB 1.1 řadič (HBA) implementující OHCI – vystupující USB sběrnici označme jako B. Na sběrnici PCI je uvnitř PIIX3 southbridge připojen další (zabudovaný) USB 1.1 řadič (HBA) implementující UHCI (poznámka: UHCI není nekompatibilní s OHCI) – vystupující USB sběrnici označme jako A. Dále je uvnitř PIIX3 na sběrnici PCI připojen (zabudovaný) řadič jednoho kanálu sběrnice Parallel ATA.

⑤ Do kořenového hubu sběrnice USB A je připojen externí 1 GB flashdisk („fleška“) HDA (používá protokol Mass Storage) s jednou partition s ext2 FS a obsahuje na sobě instalaci Linuxu. Do kořenového hubu USB B je připojena USB 1.1 klávesnice. Na jediné sběrnici Parallel ATA je připojen 4 GB harddisk HDB jako master – disk obsahuje 2 partition: 1. partition HDB1 velká 1 GB je naformátována souborovým systémem NTFS, 2. partition HDB2 je velká 3 GB a je naformátována na ext2 FS.

Otázka č. 1 (X)

Vysvětlete, jaké jsou pravděpodobné důvody k tomu, že je uvedený northbridge připojený jen k adresovým vodičům 3 až 31 na systémové sběrnici.

Otázka č. 2 (X)

Pro dané PC bychom chtěli vyrobit **ISA** kartu s N GB přídavné paměti DRAM tak, aby tato paměť byla pro aplikace přístupná podobně jako 1 GB hlavní paměti RAM (a její použití bylo pro aplikace maximálně transparentní). Pro obě následující varianty rozhodněte a vysvětlete, zda by to bylo možné, a popište, jak by v takové variantě aplikace přistoupila např. k prvnímu a jak např. k poslednímu bajtu přídavné paměti RAM:

a) N = 1 GB
b) N = 4 GB

Otázka č. 3 (X)

Předpokládejte, že implementujeme jádro nového OS, který má běžet na výše uvedeném PC, a má podporovat práci s veškerými instalovanými zařízeními. Navrhněte (nakreslete obrázek) jakým způsobem bychom v takové situaci jádro OS strukturovali. Zaměřte se na podporu aplikačního souborového API, a aplikačního API pro čtení znaků z klávesnice minimálně ve výše uvedené situaci. Pokud budete kód jádra OS rozdělovat do více modulů, tak každý takový modul označte a stručně popište jeho funkci, a vyznačte všechny ostatní moduly a části OS, se kterými bude každý takový modul komunikovat při běžném požadavku od aplikace (např. čtení nebo zápis X bytů dat z/do nějakého souboru, čtení N znaků z klávesnice). Můžete předpokládat, že náš OS poběží jen na procesorové architektuře x86. Na druhou stranu ale předpokládejte, že bychom chtěli být schopni do OS maximálně jednoduše přidávat podporu pro další zařízení, a souborové systémy.

Otázka č. 4 (X)

Po zapnutí počítače v uvedeném nastavení a konfiguraci dojde k nabootování operačního systému Linux z připojeného externího flashdisku („flešky“). Popište, co se v počítači děje od jeho zapnutí do začátku provádění instrukcí jádra Linuxu. Popište hlavně to, jaké instrukce (a kterých programů) CPU po celou dobu startu počítače zpracovává, a odkud a jakým způsobem je přečte (případně jak se instrukce na dané místo dostanou).

Otázka č. 5

Napište program v jazyce Pascal (případně v jazyce C), který na standardní výstup (tj. pomocí procedury `WriteLn`) vypíše text „`Little Endian`“ nebo „`Big Endian`“ bez uvozovek podle toho, na jaké platformě bude spuštěn (resp. pro kterou bude přeložen). Připomenutí: prefixový unární operátor @ slouží v Pascalu pro získání adresy libovolné proměnné.

Otázka č. 6

Srovnejte termíny *cluster* a *cloud* a popište jejich výhody a nevýhody.

Otázka č. 7

Předpokládejte, že máme v Pascalu funkci s následujícím prototypem:

```
function NaFixedPoint(
      hodnota : integer; exp : integer
) : word;
```

Parametry funkce reprezentují reálné číslo x následujícím způsobem: `x = hodnota * 10`^{exp}. Funkce vrací hodnotu x jako reálné číslo s pevnou desetinou čárkou ve formátu 6+10. V šestnáctkové soustavě zapište návratovou hodnotu funkce `NaFixedPoint` po zavolání s následujícími parametry (reprezentujícími hodnotu `3,875`):

```
… := NaFixedPoint(3875, -3);
```

Předpokládejte, že typ `integer` je běžná Pascalová reprezentace celého 16-bitového znaménkového (signed) čísla, typ `word` je běžná Pascalová reprezentace celého 16-bitového bezznaménkového (unsigned) čísla.

Otázka č. 8

Předpokládejte, že v Pascalu implementujete část operačního systému s podporou pro vícevláknové zpracování a s preemptivním přepínáním vláken. Předpokládejte, že OS podporuje víceprocesorové SMP systémy, a vždy poběží na počítači s minimálně dvěma procesory (nebo procesorovými jádry). Vaším úkolem je naimplementovat podporu pro speciální druh zámků, tzv. *spinlock*, jehož využití je na daném systému někdy výhodné. *Spinlock* je běžný zámek ale s aktivním čekáním. Vaše implementace nemusí podporovat rekurzivní zamykání. Vaším úkolem je tedy v Pascalu naimplementovat následující záznam a k němu náležející procedury.
K dispozici máte fci CAS, která je standardní implementací operace *Compare & Swap* (resp. *Compare & Exchange*). Popište její chování a vhodně ji využijte ve vaší implementaci.

```
type
    TSpinLock = record
        ...
    end;
    PSpinLock = ^TSpinLock;

procedure Lock(lock : PSpinLock);
procedure Unlock(lock : PSpinLock);
```

Otázka č. 9

Předpokládejme, že implementujeme operační systém poskytující podporu pro vícevláknové zpracování, dynamicky linkované knihovny a stránkování (tj. běží jen na procesorových platformách s podporou stránkování). Předpokládejme situaci, že více procesů běžících v takovém systému často používá stejnou dynamicky linkovanou knihovnu. Abychom ušetřili paměť, tak bychom chtěli, aby v takovém případě byl minimálně kód knihovny uložený v paměti RAM jen jednou. Je možné to nějak zařídit? Pokud ano, popište přesně, jakým způsobem by to bylo možné, a jak bude v takovém systému probíhat spouštění aplikace používající takovou „sdílenou“ DLL. Pokud ne, popište přesně všechny důvody, proč není možné takový systém naimplementovat.

Otázka č. 10

Předpokládejte, že v počítači máme přes 32-bit sběrnici PCI připojený řadič GPIO (General Purpose Input/Output), který zpřístupňuje 32 nezávislých digitálních výstupních signálů/linek (tzv. GPIO). Řadič je nakonfigurovaný tak, že má od adresy `0x4000` v paměťovém adresovém prostoru namapovaný jeden 32-bitový port, kde každý z jeho bitů reprezentuje stav jednoho GPIO výstupního signálu. Čtením z tohoto portu zjistíme aktuální stav signálů, které řadič „vysílá“; zápisem nastavíme novou hodnotu signálů, které řadič „vysílá“. Vaším úkolem je v Pascalu s vhodným typickým rozšířením (např. Free Pascal) naimplementovat proceduru s následujícím prototypem:

```
procedure Output(b : Longword);
```

kde typ `Longword` je 32-bit celočíselný typ bez znaménka, a platná hodnota pro `b` je libovolné číslo `0` až `255`. Proměnná `b` se skládá z následujících částí: bity 2 až 7 reprezentují znaménkové 6-bitové celé číslo C **ve dvojkovém doplňku**, bit 0 je příznakem pojmenovaným `F0`, bit 1 je příznakem `F1`.
Účelem této procedury je **najednou (v jednom okamžiku)** změnit stav GPIO signálů „vysílaných“ GPIO řadičem následujícím způsobem (vše počítáno od `0`):

- 0. až 5. GPIO na absolutní hodnotu čísla C
- 29. GPIO na `0`, když je `F0` rovno `1`; a na `1`, když je `F0` rovno `0`
- 30. GPIO na hodnotu `F1`
- 31. GPIO na `0`
- Ostatní GPIO linky musí zůstat nezměněné!

Váš kód vždy poběží na počítači s procesorem Intel Pentium III (32-bitový procesor, 64-bit datová sběrnice, 36-bit adresová sběrnice, separátní 16-bit I/O adresový prostor). Pořadí bitů i bytů chápané procesorem, sběrnicí PCI, i řadičem GPIO jsou stejná. Předpokládejte, že stránkování, ani segmentace nejsou zapnuté.

Pozor: instrukce `MOV` má u procesoru Intel Pentium III kromě varianty s **32-bit** operandy i varianty s **8-bit, resp. 16-bit** operandy, které zde ale **nesmíme** použít, přestože na první pohled vypadá jejich použití jako ekvivalentní – protože např. čtyři postupné 8-bitové zápisy na adresy `0x4000`, `0x4001`, `0x4002`, `0x4003` nejsou pro nás ekvivalentní jednomu 32-bitovému zápisu na adresu `0x4000`, protože **způsobí jen postupnou změnu** potřebných linek GPIO. Máte ale zaručeno, že použitý překladač Pascalu vždy generuje instrukci `MOV` s 32-bitovým operandem, pokud do paměti zapisujete (nebo čtete) zarovnanou 32-bitovou hodnotu!