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).

Otázka č. 1

Předpokládejte systém s preemtivním plánováním vláken, kde každé vlákno má přidělenou nějakou pevnou prioritu v rozsahu 0-31 (0 je nejvyšší priorita, 31 je nejnižší priorita). Systém aplikacím též poskytuje standardní implementaci zámků (mutexů). Předpokládejte, že na takovém systému spustíme 2 vlákna, která provádí následující posloupnosti operací (předpokládejte, že `lock X` je zamčení zámku *X*, `unlock X` je odemčení zámku *X*, `opShort` je nějaká krátce trvající operace [trvá v řádu jednotek taktů procesoru], `opLong` je nějaká dlouho trvající operace [trvá v řádu jednotek sekund]):

vlákno *T1* (priorita 14):

```
lock A
opShort
unlock A
lock B
opLong
lock A
opLong
unlock A
opLong
unlock B
```

vlákno *T2* (priorita 15):

```
lock C
lock A
lock B
opShort
unlock C
unlock B
unlock A
```

V tomto kontextu odpovězte na následující: Může někdy dojít k problému zvanému *deadlock*? Pokud ano, popište alespoň jeden takový případ. Pokud ne, vysvětlete proč.

Otázka č. 2

Naimplementujte v Pascalu funkci `DelkaTextu` s následujícím prototypem:

```
type
  PUtf8 = ^byte;
function DelkaTextu(text : PUtf8) : word;
```

která jako parametr `text` bere ukazatel na null-terminated řetězec v kódování UTF-8, a vrátí počet kompletních znaků (grafémů), ze kterých se tento řetězec skládá (tedy např. pro libovolný vstupní řetězec reprezentující text `Říp` má funkce `DelkaTextu` vrátit hodnotu 3). Předpokládejte, že velikost typu `word` jsou 2 byty, velikost typu `byte` je 1 byte. Pokud byste od RTL nutně potřebovali nějaké pomocné funkce nebo procedury, tak si je zadeklarujte, a popište chování, které od nich očekáváte. **Pozor:** vstupní řetězec `text` může být opravdu libovolný validní text v UTF-8!

Otázka č. 3

CIL kód je strojový kód virtuálního stroje se zásobníkovou architekturou (zásobníkového stroje). CIL kód má load/store architekturu. CIL kód má následující instrukce:

- `LDSFLD` *`adresa`* – načtení hodnoty na zadané adrese (argument instrukce)
- `STSFLD` *`adresa`* – uložení hodnoty na zadanou adresu (argument instrukce)
- `LDC` *`číslo`* – načtení celočíselné konstanty (argument instrukce)
- `ADD` – sečtení dvou čísel (instrukce bez explicitních argumentů)
- `MUL` – vynásobení dvou čísel (instrukce bez explicitních argumentů)

Předpokládejte, že registrový zásobník má neomezenou hloubku. Přepište následující výraz v Pascalu do ekvivalentní posloupnosti instrukcí strojového kódu CIL (proměnné A, B, C jsou uloženy na následujících adresách: A = `$5000`, B = `$6000`, C = `$7000`):

```
A := (A * B) + C + (16 * A)
```

Otázka č. 4

Předpokládejte programové prostředí, které poskytuje podporu pro vícevláknové zpracování. Předpokládejte, že hlavní procedura nějakého vlákna doběhne do konce (tj. toto vlákno je ve stavu těsně před provedením instrukce RET [instrukce návratu z podprogramu] na konci jeho hlavní procedury). Popište a vysvětlete, co se bude dít potom (jaký kód bude procesor dále zpracovávat a co tento kód bude dělat).

Otázka č. 5

Předpokládejte, že máme aplikaci A, která pro komunikaci s OS používá POSIX API verze 1b. Aplikaci A máme přeloženou tak, že výsledný binární spustitelný soubor XX jsme schopni bez problémů spustit na běžné instalaci OS Linux (který též implementuje POSIX.1b API). Dále víme, že existuje SW abstrakční vrstva Cygwin nad Win32 API pro Windows řady NT, a máme verzi Cygwinu, který implementuje všechny procedury a funkce z POSIX.1b API, které používá aplikace A. Jsou výše uvedené informace dostatečné pro to, abychom rozhodli, zda jsme pomocí Cygwin schopni spustit binární spustitelný soubor XX na Windows 7 (řada NT)? Pokud ano, vysvětlete proč. Pokud ne, napište všechny informace, které nám pro rozhodnutí schází, a vysvětlete proč.

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 V 1. PCI slotu je vložena grafická karta S3 Trio V64. Ve 2. PCI slotu je vložena SCSI [čti “skazi”] karta Adaptec AHA-2940 (s HBA [řadičem] AIC-7850). Ke kartě je po sběrnici SCSI (1. SCSI sběrnice v počítači) připojen pevný disk HDA s jednou partition s FAT32 FS, a pevný disk HDB s jednou partition s ext2 FS. V 1. ISA slotu je vložena SCSI karta Adaptec AVA-1505 (s HBA [řadičem] AIC-6260). Ke kartě je po SCSI sběrnici (2. SCSI sběrnice v počítači) připojen scanner HP ScanJet 4P. Řadiče AIC-7850 a AIC-6260 mají rozdílné vzájemně nekompatibilní HCI.

SCSI je paralelní multidrop sběrnice. Po sběrnici SCSI se data přenášejí ve formě paketů. Formát SCSI paketů je standardizovaný. Payload každého takového paketu tvoří příkazy připojeným zařízením a odpovědi na ně. Pro každý typ zařízení připojitelného na sběrnici SCSI (disk nebo CD-ROM mechanika nebo scanner) existuje standardizovaná množina a formát takových příkazů.

Otázka č. 6 (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. 7 (X)

Pro dané PC bychom chtěli vyrobit PCI 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 = 2 GB
b) N = 8 GB

Otázka č. 8 (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 obrázku ze scannerů 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 Y bytů obrazových dat ze scanneru). Dále bychom chtěli, aby náš OS byl jednoduše na úrovni zdrojových kódů přenositelný i na jinou procesorovou architekturu než x86.

Otázka č. 9

Následující obrázek obsahuje část screenshotu hex editoru, který zobrazuje obsah 68 bytů dlouhého binárního souboru:

	0001	0203	0405	0607	0809	0A0B	0C0D	0E0F
00	0000	0000	0000	F07F	0000	0000	0000	F0FF
10	5000	5901	ED00	6C00	6900	6101	2000	7E01
20	6C00	7500	6501	6F00	7500	0D01	6B00	FD00
30	2000	6B00	6F01	4801	2E00	E000	80C7	7200
40	E100	6400						

Víme, že všechna data jsou v souboru uložena jako little endian, a že od 33. bytu (počítáno od 0) je v souboru uloženo 16-bitové reálné číslo s pevnou desetinnou čárkou ve formátu 5+11. Zapište hodnotu tohoto reálného čísla v desítkové soustavě.

Otázka č. 10

Předpokládejte provedení následující instrukce:

```
MOV   EAX, [2046]
```

která načítá 4 bytovou hodnotu z adresy 2046 do registru EAX. Operační kód instrukce začíná na adrese 0xFFFE a její celková délka je 6 bytů. Systém používá stránky o velikosti 1 kB a jednoúrovňové stránkovací tabulky. Rozhodněte, kolik výpadků stránky (page faults) může celkem maximálně vzniknout v průběhu zpracování této instrukce. Popište proč. Očekávejte, že při každém výpadku stránky dojde k obnovení mapování dané stránky a v rámci zpracování dané instrukce již k dalšímu výpadku stejné stránky nedojde.