Mark: 26/30 (total score: 26/30)

+15/1/4+

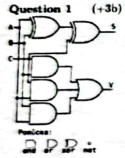
Architektura počítačů (B0B35APO) **Zkouškový test, LS2023/24**

← Prosíme, vyplňte zde své šestimístné identifikační číslo (uvedeno na kartě ISIC). Toto číslo bude přednostně použito pro Vaši identifikaci. Zároveň uveďte své jméno a příjmení.

Jméno a příjmení:

Čas na vypracování testu: 90 minut. Otázky označené symbolem ♣ mohou mít vícero správných odpovědí. Ostatní otázky mají pouze jednu správnou odpověď.

Question 1 (+3b) Vyberte co dělá logický obvod na tomto obrázku:



3/3
- ☐ realizace registru obvodem d-latch
- ☐ poloviční sčítačka (half-adder)
- ☒ plná sčítačka (full-adder)
- ☐ multiplexor

Question 2 (±1b) Při reprezentaci reálného čísla 32 bity podle normy IEEE-754 je 5 bitů určených pro exponent?

1/1 ☒ NE ☐ ANO

Question 3 (±1b) Existuje paměť cache, která má velikost řádku 11 bajtů?

0/1 ☒ NE ☐ ANO

Question 4 (±1b) Použití pipeliningu zrychluje přístup procesoru do paměti oproti jednocyklové verzi?

0/1 ☐ ANO ☒ NE

Question 5 (±1b) PCIe (PCI express) je příkladem sériové sběrnice?

1/1 ☐ NE ☒ ANO

Question 6 (±1b) Při volání funkce v RISC-V se návrtová adresa uloží do vybraného registru?

1/1 ☒ ANO ☐ NE

Question 7
(+3b) Pokud následující 6-bitové binární číslo 111110 je kódováno v dvojkovém doplňku, kterému dekadickému číslu odpovídá?

3/3

☐ -34	☐ 62	☐ 46	☒ -2	☐ 5
☐ 30	☐ 35	☐ 39	☐ 29	☐ 27
☐ 33	☐ 20	☐ 23	☐ 10	☐ -1

+15/2/3+

Zcela jistě znáte způsob reprezentace čísel v pohyblivé řádové čárce v jednoduché přesnosti dle IEEE745. Nicméně Vaše aplikace vyžaduje, aby číslo v plovoucí řádové čárce zabíralo pouze 9 bitů. Proto jste navrhli reprezentaci, ve které má exponent 3 bity, mantisa 5 bitů. Jinak chcete zachovat všechny vlastnosti IEEE754.

Question 8 (+2b) Jak se binárně zakóduje dekadické číslo 3.71 v této reprezentaci? (**Pokyny pro vyplnění:** Box zaškrtněte pouze pokud má být daný bit nastaven na 1. Prázdný box je považován za 0. Nejméně významný bit je úplně vpravo. I pokud vychází zbytek větší než 0.5 LSB tak mantisu nahoru nezaokrouhlujte.)

2/2 ☐1 ☒1 ☐1 ☐1 ☒1 ☒1 ☐1 ☒1 ☒1

Question 9 (+1b) Jak bude v této reprezentaci binárně zakódováno největší možné zobrazitelné reálné číslo?

1/1 ☐1 ☒1 ☒1 ☐1 ☒1 ☒1 ☒1 ☒1 ☒1

Question 10 (+1b) Jaká bude dekadická hodnota tohoto největšího možného čísla? Uvažujte 3 desetinné pozice.

1/1

☐ 430.500	☐ 54.125	☐ 18.375	☒ 15.750	☐ 147.500	☐ 25.000
☐ 216.000	☐ 50.250	☐ 26.875	☐ 43.000	☐ 216.250	☐ 146.500

Question 11 (+1b) Jak bude v této reprezentaci zakódováno nejmenší nenulové kladné normalizované číslo?

1/1 ☐1 ☐1 ☐1 ☒1 ☐1 ☐1 ☐1 ☐1 ☐1

Question 12 (+1b) Jaká bude dekadická hodnota tohoto nejmenšího nenulového kladného normalizovaného čísla? Zaokrouhlete na 3 desetinné pozice.

1/1

☐ 0.117	☐ 0.008	☐ 0.125	☐ 0.094	☐ 0.000	☐ 0.059
☐ 0.156	☐ 0.010	☐ 0.063	☐ 0.025	☒ 0.250	☐ 0.047

Uveďte strojový kód instrukce "add" (v hexadecimálním tvaru) níže uvedeného fragmentu. Pokud některá z položek instrukce není specifikována nebo ze zadání úkolu nevyplývá, pak pro účely tohoto zkouškového testu může daná položka nabývat libovolnou hodnotu – ideálně je však tuto položku nevyplňovat. Operační kód použitých instrukcí je následující: addi.0b0010011, add.0b0110011, sw.0b0100011. Instrukce addi má fct3 = 0, instrukce add má funct7=0 a fct3 = 0, instrukce sw má fct3= 0b010.

```
addi x19, x19, -4
add x22, x22, x19
sw x22,-4(x7)
```

I	imm(12) 31:20		rs1(5), 19:15	fct3,14:12	rd(5),11:7	opcode(7),6:0
R	funct7(7) 31:25	rs2(5), 24:20	rs1(5), 19:15	fct3,14:12	rd(5),11:7	opcode(7),6:0
S	imm[11:5], 31:25	rs2(5), 24:20	rs1(5), 19:15	fct3,14:12	imm[4:0],11:7	opcode(7),6:0

Question 13 (+2b) Jak bude zakódována horní polovina instrukce "add" v hexadecimálním tvaru? Tzn. horních 16 bitů instrukce vyjádřené v strojovém kódu. (První cifru čísla vyplňte do prvního řádku, druhou cifru do druhého řádku atd. Hodnotám 10, 11, 12, 13, 14 a 15 odpovídá A, B, C, D, E a F.

2/2

☒0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☐0	☒1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☒3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☒11	☐12	☐13	☐14	☐15

Question 14 (+2b) Jak bude zakódována dolní polovina instrukce "add" v hexadecimálním tvaru?

2/2

☒0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☒11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☒3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☒3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15

Skenováno pomocí CamScanner

Question 15 (+1b) Datový typ float, který se používá v jazyce C se v paměti obvykle ukládá v:

- ☐ Hammingově kódu
- ☐ aditivním kódu
- ☐ Grayově kódu
- ☐ přímém kódu
- ☐ BCD kódu
- ☒ dle standardu IEEE 754
- ☐ dvojkovém doplňku
- ☐ inverzním kódu
- ☐ není specifikováno

Question 16
(+1b) Algoritmus LFU v případě konfliktu platných řádek v cache nahrazuje

- ☐ naposled přidaný řádek
- ☐ náhodně vybraný řádek
- ☐ naposled použitý řádek
- ☐ řádek s nejnižší hodnotou
- ☐ nejdéle nepoužitý řádek
- ☐ řádek s nejnižší adresou
- ☒ nejméně používaný řádek
- ☐ dosud nepoužitý řádek
- ☐ první/nejstarší přidaný řádek

Question 17
(+2b) Pokud uvažujeme desáteckové číslo 14 a aditivní konstantu 15, jaká bude 6-bitová binární reprezentace takto kódovaného čísla v kódu s posunutou nulou?

☐ 100111	☐ 100011	☐ 010111	☐ 111001	☐ 111010
☐ 001101	☐ 000111	☐ 110110	☐ 101010	☐ 111011
☒ 011101	☐ 110100	☐ 010110	☐ 010001	☐ 111100

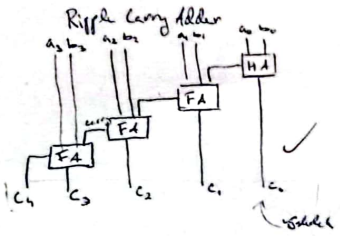
Question 18 (+5b) Nakreslete pomalou 4-bitovou sčítačku s postupným výpočtem carry (řetězová sčítačka – Ripple Carry Adder) pro součet dvou čísel $a = a_3, a_2, a_1, a_0$ a $b = b_3, b_2, b_1, b_0$ v dvojkovém doplňku. V obrázku vyznačte výsledek, číslo $c = c_4, c_3, c_2, c_1, c_0$
K dispozici máte:

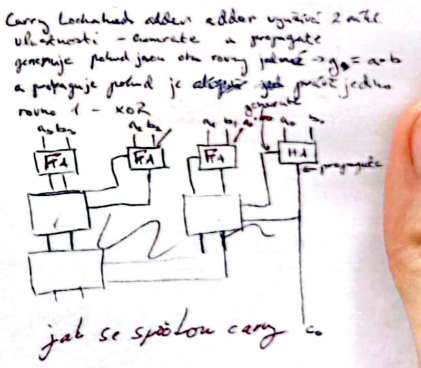
1. poloviční sčítačku (Half Adder) s jednobitovými vstupy a,b a výstupy c_{out},s (dvoubitové číslo $c_{out}s$), která realizuje $c_{out}s = a + b$

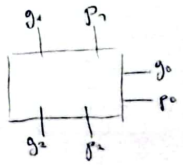
2. úplnou sčítačku (Full Adder) s jednobitovými vstupy a,b,c a výstupy c_{out},s (dvoubitové číslo $c_{out}s$), která realizuje $c_{out}s = a + b + c$

Dále nakreslete rychlou sčítačku Carry Lookahead adder, která použije stejné komponenty (FA, HA), ale může využít na vstupu libovolnou logickou funkci, např. $a_2 + b_2 \cdot (a_1 + a_0)$, kde operace + je logický součet or, operace · je logický součin and. Pro přehlednost si můžete nadefinovat i pomocné mezivýpočty, např. $g_0 = a_0 \cdot b_0$, $p_1 = a_1 + b_1$. Pokud nezvládnete celou sčítačku, nakreslete alespoň část.

☐w ☐p1 ☐p2 ☒p3 ☐p4 ☐c 3/5







$g_2 = g_1 \cdot g_0 + g_0 p_1 +$

$p_2 = p_1 \cdot p_0 +$

Skenováno pomocí CamScanner

Mark: [illegible] (total score: 25/30)

•25/1/24•

Architektura počítačů (B0B35APO) **Zkouškový test, LS2023/24**

← *Prosíme, vyplňte zde své šestimístné identifikační číslo (uvedeno na kartě ISIC). Toto číslo bude přednostně použito pro Vaši identifikaci. Zároveň uveďte své jméno a příjmení.*

Čas na vypracování testu: 90 minut. Otázky označené symbolem & mohou mít více správných odpovědí. Ostatní otázky mají pouze jednu správnou odpověď.

Question 1 (+3b) Vyberte co dělá logický obvod na tomto obrázku:

3/3
- ☐ poloviční sčítačka (half-adder)
- ☐ plná sčítačka (full-adder)
- ☐ realizace registru obvodem d-latch
- ☒ multiplexor

Question 2 (±1b) K číslu X v dvojkovém doplňku dostaneme číslo -X také v dvojkovém doplňku tak, že provedeme bitovou negaci všech bitů X a následně přičteme 1?

1/1
- ☐ NE
- ☒ ANO

Question 3 (±1b) Může existovat 3-cestná paměť cache (cache se stupněm asociace 3)?

0/1
- ☐ NE
- ☒ ANO

Question 4 (±1b) Datový hazard nastane, pokud jedna instrukce do registru zapíše a následující instrukce z registru čte?

1/1
- ☐ NE
- ☒ ANO

Question 5 (±1b) Sériové sběrnice nemohou být full-duplex (plný duplex), paralelní sběrnice mohou?

1/1
- ☒ NE
- ☐ ANO

Question 6 (±1b) Při použití výjimek program přímo vykoná na opakované dotazování se stavu periferií?

1/1
- ☒ NE
- ☐ ANO

Question 7
(+3b) Pokud následující 6-bitové binární číslo 100001 je kódováno v dvojkovém doplňku, kterému dekadickému číslu odpovídá?

3/3

☐ -24	☐ 1	☐ 0	☐ 4	☐ 6
☐ 33	☐ -9	☐ 10	☐ 17	☐ -30
☐ -2	☐ -6	☒ -31	☐ -19	☐ -63

•25/1/23•

Zcela jistě znáte způsob reprezentace čísel v pohyblivé řádové čárce v jednoduché přesnosti dle IEEE754. Navrhněte Vaše aplikace vyžaduje, aby číslo v plovoucí řádové čárce zabíralo pouze 9 bitů. Proto jste navrhli reprezentaci, ve které má exponent 4 bity, mantisa 4 bity. Jinak chcete zachovat všechny vlastnosti IEEE754.

Question 8 (+2b) Jak se binárně zakóduje desetinné číslo 0.91 v této reprezentaci? (Pokyny pro vyplnění: Bity znázorňujte pouze pokud má být daný bit nastaven na 1. Případný bod je považován za 0. Nejméně významný bit je úplně vpravo. V případě vyskání zbytek větší než 0.5 LSB tak mantisu zaokrouhlujte.)

2/2

Question 9 (+1b) Jak bude v této reprezentaci binárně zakódováno nejmenší možné zobrazitelné reálné číslo?

1/1

Question 10 (+1b) Jaká bude dekadická hodnota tohoto nejvyššího možného čísla? Uvažujte 3 desetinná místa.

1/1

☐ 117.500	☐ 29.000	☐ 32.000	☐ 59.375	☐ 390.000	☐ 201.750
☐ 94.125	☒ 248.000	☐ 369.250	☐ 171.500	☐ 4.375	☐ 3.375

Question 11 (+1b) Jak bude v této reprezentaci zakódováno nejmenší nenulové kladné normalizované číslo?

1/1

Question 12 (+1b) Jaká bude dekadická hodnota tohoto nejmenšího nenulového kladného normalizovaného čísla? Zaokrouhlete na 3 desetinné pozice.

1/1

☐ 0.023	☒ 0.016	☐ 0.047	☐ 0.188	☐ 1.000	☐ 0.182
☐ 0.166	☐ 0.172	☐ 0.000	☐ 0.063	☐ 0.000	☐ 0.102

Uveďte strojový kód instrukce "addi" (v hexadecimálním tvaru) níže uvedeného fragmentu. Pokud některá z polí instrukce není specifikována nebo ze zadání úkolu nevyplývá, pak pro účely tohoto zkouškového testu může daná položka nabývat libovolnou hodnotu – ideálně je však tuto položku nevyplňovat. Operační kód použitých instrukcí je následující (addi: 0b0010011, add:0b0110011, sw:0b0100011. Instrukce addi má fct3 = 0, instrukce add má funct7=0 a fct3 = 0, instrukce sw má fct3 = 0b010

addi x7, x7, -12
add x10, x10, x7
sw x10,-12(x20)

I	imm[12] 31:20		rs1(5), 19:15	fct3,14:12	rd(5),11:7	opcode(7),6:0
R	funct7(7) 31:25	rs2(5), 24:20	rs1(5), 19:15	fct3,14:12	rd(5),11:7	opcode(7),6:0
S	imm[11:5], 31:25	rs2(5), 24:20	rs1(5), 19:15	fct3,14:12	imm[4:0],11:7	opcode(7),6:0

Question 13 (+2b) Jak bude zakódována horní polovina instrukce "addi" v hexadecimálním tvaru? Tzn. horních 16 bitů instrukce vyjádřené v strojovém kódu. (První cifru čísla vypište do prvního řádku, druhou cifru do druhého řádku atd. Hodnotám 10, 11, 12, 13, 14 a 15 odpovídá A, B, C, D, E a F.

0/2

Question 14 (+2b) Jak bude zakódována dolní polovina instrukce "addi" v hexadecimálním tvaru?

0/2

Skenováno pomocí CamScanner

+25/3/22+

Question 15 (+1b) Datový typ int, který se používá v jazyce C se v paměti obvykle ukládá v:

1/1

- ☐ Hammingově kódu
- ☐ BCD kódu
- ☐ inverzním kódu
- ☐ není specifikováno
- ☐ přímém kódu
- ☐ aditivním kódu
- ☐ Grayově kódu
- ☐ dle standardu IEEE 754
- ☒ dvojkovém doplňku

Question 16
(+1b) Algoritmus LFU v případě konfliktu platných řádek v cache nahrazuje

1/1

- ☐ naposled použitý řádek
- ☐ nejdéle nepoužitý řádek
- ☐ naposled přidaný řádek
- ☐ řádek s nejnižší hodnotou
- ☐ řádek s nejnižší adresou
- ☐ dosud nepoužitý řádek
- ☒ nejméně používaný řádek
- ☐ první/nejstarší přidaný řádek
- ☐ náhodně vybraný řádek

Question 17
(+2b) Pokud uvažujeme dekadické číslo 9 a aditivní konstantu 31, jaká bude 6-bitová binární reprezentace takto kódovaného čísla v kódu s posunutou nulou?

2/2

- ☐ 001111
- ☐ 000001
- ☐ 000011
- ☐ 110010
- ☐ 100010
- ☐ 011111
- ☒ 101000
- ☐ 001110
- ☐ 111000
- ☐ 100100
- ☐ 001001
- ☐ 010110
- ☐ 100011
- ☐ 010010
- ☐ 001100

+25/4/21+

Question 18 (+5b) Nakreslete pomalou 4-bitovou sčítačku s postupným výpočtem carry (řetězová sčítačka – Ripple Carry Adder) pro součet dvou čísel $a = a_3, a_2, a_1, a_0$ a $b = b_3, b_2, b_1, b_0$ v dvojkovém doplňku. V obrázku vyznačte výsledek, číslo $c = c_4, c_3, c_2, c_1, c_0$
K dispozici máte:

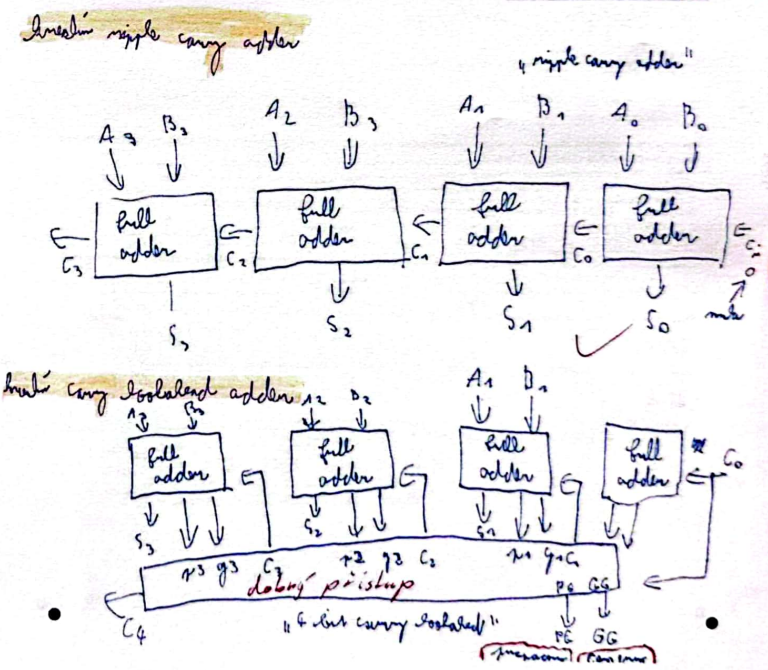
1. poloviční sčítačku (Half Adder) s jednobitovými vstupy a,b a výstupy c_o, s (dvoubitové číslo $c_o s$), která realizuje $c_o s = a + b$

2. úplnou sčítačku (Full Adder) s jednobitovými vstupy a,b,c a výstupy c_o, s (dvoubitové číslo $c_o s$), která realizuje $c_o s = a + b + c$

Dále nakreslete rychlou sčítačku Carry Lookahead adder, která použije stejné komponenty (FA, HA), ale může využít na vstupu libovolnou logickou funkci, např. $a_2 + b_2 \cdot (a_1 + a_0)$, kde operace + je logický součet or; operace · je logický součin and. Pro přehlednost si můžete nadefinovat i pomocné mezivýpočty, např. $g_0 = a_0 \cdot b_0$, $p_1 = a_1 + b_1$. Pokud nezvládnete celou sčítačku, nakreslete alespoň část.

☐ w ☐ p1 ☐ p2 ☐ p3 ☐ p4 ☒ c

5/5



Skenováno pomocí CamScanner

Mark: 25/30 (total score: 25/30)

+26/1/20+

Architektura počítačů (B0B35APO) **Zkouškový test, LS2023/24**

← *Prosíme, vyplňte zde své šestimístné identifikační číslo (uvedeno na kartě ISIC). Toto číslo bude přednostně použito pro Vaši identifikaci. Zároveň uveďte své jméno a příjmení.*

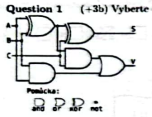
Jméno příjmení:

.....

.....

Čas na vypracování testu: 90 minut. Otázky označené symbolem ♣ mohou mít vícero správných odpovědí. Ostatní otázky mají pouze jednu správnou odpověď.

Question 1 (+3b) Vyberte co dělá logický obvod na tomto obrázku:



3/3
- ☐ realizace registru obvodem d-latch
- ☐ multiplexor
- ☐ poloviční sčítačka (half-adder)
- ☒ plná sčítačka (full-adder)

Question 2 (±1b) Lze operaci not sestrojit pomocí hradla nor?

1/1 ☒ ANO ☐ NE

Question 3 (±1b) Procesor RISC-V může pracovat s pamětí i v instrukci lw?

1/1 ☒ ANO ☐ NE

Question 4 (±1b) Použití pipeliningu zrychluje přístup procesoru do paměti oproti jednocyklové verzi?

1/1 ☐ ANO ☒ NE

Question 5 (±1b) Sériové sběrnice mohou být full-duplex (plný duplex), paralelní sběrnice nemohou?

1/1 ☒ NE ☐ ANO

Question 6 (±1b) Při volání funkce v RISC-V se první argument uloží do kódu volané funkce?

1/1 ☒ NE ☐ ANO

Question 7

(+3b) Pokud následující 6-bitové binární číslo 101101 je kódováno v dvojkovém doplňku, kterému dekadickému číslu odpovídá?

3/3
- ☐ 13
- ☐ 6
- ☐ -7
- ☐ -51
- ☐ 10
- ☐ 3
- ☐ -12
- ☐ 18
- ☐ 12
- ☐ 29
- ☐ 22
- ☐ -18
- ☐ 16
- ☒ -19
- ☐ 45

+26/2/19+

Zcela jistě znáte způsob reprezentace čísel v pohyblivé řádové čárce v jednoduché přesnosti dle IEEE745. Nicméně Vaše aplikace vyžaduje, aby číslo v plovoucí řádové čárce zabíralo pouze 10 bitů. Proto jste navrhli reprezentaci, ve které má exponent 3 bity, mantisa 6 bitů. Jinak chcete zachovat všechny vlastnosti IEEE754.

Question 8 (+2b) Jak se binárně zakóduje dekadické číslo 3.50 v této reprezentaci? (**Pokyny pro vyplnění:** Box **zaškrtkujte** pouze pokud má být daný bit nastaven na 1. Prázdný box je považován za 0. Nejméně významný bit je úplně vpravo. I pokud vychází zbytek větší než 0.5 LSB tak mantisu nahoru nezaokrouhlujte.)

2/2 ☐1 ☒1 ☐1 ☐1 ☒1 ☒1 ☐1 ☐1 ☐1 ☐1

Question 9 (+1b) Jak bude v této reprezentaci binárně zakódováno největší možné zobrazitelné reálné číslo?

1/1 ☐1 ☒1 ☒1 ☐1 ☒1 ☒1 ☒1 ☒1 ☒1 ☒1

Question 10 (+1b) Jaká bude dekadická hodnota tohoto největšího možného čísla? Uvažujte 3 desetinné pozice.

1/1
- ☐ 74.500
- ☐ 132.500
- ☐ 46.000
- ☐ 28.500
- ☐ 8.375
- ☐ 1.250
- ☐ 36.750
- ☐ 179.000
- ☐ 23.625
- ☒ 15.875
- ☐ 142.625
- ☐ 12.375

Question 11 (+1b) Jak bude v této reprezentaci zakódováno nejmenší nenulové kladné normalizované číslo?

1/1 ☐1 ☐1 ☐1 ☒1 ☐1 ☐1 ☐1 ☐1 ☐1 ☐1

Question 12 (+1b) Jaká bude dekadická hodnota tohoto nejmenšího nenulového kladného normalizovaného čísla? Zaokrouhlete na 3 desetinné pozice.

1/1
- ☐ 0.750
- ☐ 0.158
- ☐ 0.203
- ☐ 0.131
- ☐ 0.021
- ☐ 0.063
- ☐ 0.027
- ☐ 0.000
- ☐ 0.031
- ☒ 0.250
- ☐ 0.000
- ☐ 0.055

Uveďte strojový kód instrukce "sw" (v hexadecimálním tvaru) níže uvedeného fragmentu. Pokud některá z pozic instrukce není specifikována nebo ze zadání úkolu nevyplývá, pak pro účely tohoto zjišťovacího testu může daná pozice nabývat libovolnou hodnotu – ideálně je však tuto položku nevyplňovat. Operační kód použitých instrukcí je následující: addi:0b0010011, add:0b0110011, sw:0b0100011. Instrukce addi má fct3 = 0, instrukce add má funct7=0 a fct3 = 0, instrukce sw má fct3= 0b010.

```
addi x5, x5, -12
add x8, x8, x5
sw x8,-12(x16)
```

I	imm[12] 31:20		rs1(5), 19:15	fct3,14:12	rd(5),11:7	opcode(7),6:0
R	funct7(7) 31:25	rs2(5), 24:20	rs1(5), 19:15	fct3,14:12	rd(5),11:7	opcode(7),6:0
S	imm[11:5], 31:25	rs2(5), 24:20	rs1(5), 19:15	fct3,14:12	imm[4:0],11:7	opcode(7),6:0

Question 13 (+2b) Jak bude zakódována horní polovina instrukce "sw" v hexadecimálním tvaru? Tzn. horních 16 bitů instrukce vyjádřené v strojovém kódu. (První cifru čísla vyplňte do prvního řádku, druhou cifru do druhého řádku atd. Hodnotám 10, 11, 12, 13, 14 a 15 odpovídá A, B, C, D, E a F.

2/2
- ☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10 ☐11 ☐12 ☐13 ☐14 ☒15
- ☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10 ☐11 ☐12 ☐13 ☒14 ☐15
- ☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☒8 ☐9 ☐10 ☐11 ☐12 ☐13 ☐14 ☐15
- ☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☒8 ☐9 ☐10 ☐11 ☐12 ☐13 ☐14 ☐15

Question 14 (+2b) Jak bude zakódována dolní polovina instrukce "sw" v hexadecimálním tvaru?

2/2
- ☐0 ☐1 ☒2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10 ☐11 ☐12 ☐13 ☐14 ☐15
- ☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☒10 ☐11 ☐12 ☐13 ☐14 ☐15
- ☐0 ☐1 ☒2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10 ☐11 ☐12 ☐13 ☐14 ☐15
- ☐0 ☐1 ☐2 ☒3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9 ☐10 ☐11 ☐12 ☐13 ☐14 ☐15

Skenováno pomocí CamScanner

Question 15 (+1b) Datový typ int, který se používá v jazyce C se v paměti obvykle ukládá v:

1/1

- ☐ přímém kódu
- ☐ aditivním kódu
- ☐ BCD kódu
- ☐ inverzním kódu
- ☐ Hammingově kódu
- ☐ není specifikováno
- ☐ Grayově kódu
- ☐ dle standardu IEEE 754
- ☒ dvojkovém doplňku

Question 16
(+1b) Algoritmus LFU v případě konfliktu platných řádek v cache nahrazuje

1/1

- ☐ dosud nepoužitý řádek
- ☐ řádek s nejnižší adresou
- ☐ řádek s nejnižší hodnotou
- ☐ naposled použitý řádek
- ☐ nejdéle nepoužitý řádek
- ☐ naposled přidaný řádek
- ☒ nejméně používaný řádek
- ☐ náhodně vybraný řádek
- ☐ první/nejstarší přidaný řádek

Question 17
(+2b) Pokud uvažujeme dekadické číslo 37 a aditivní konstantu 15, jaká bude 6-bitová binární reprezentace takto kódovaného čísla v kódu s posunutou nulou?

0/2

- ☐ 111000
- ☐ 010000
- ☒ 010011
- ☐ 111111
- ☐ 001010
- ☐ 011111
- ☐ 100100
- ☒ 110100
- ☐ 000011
- ☐ 011101
- ☐ 001110
- ☐ 010101
- ☐ 101110
- ☐ 111110

Question 18 (+5b) Nakreslete pomalou 4-bitovou sčítačku s postupným výpočtem carry (řetězová sčítačka – Ripple Carry Adder) pro součet dvou čísel $a = a_3, a_2, a_1, a_0$ a $b = b_3, b_2, b_1, b_0$ v dvojkovém doplňku. V obrázku vyznačte výsledek, číslo $c = c_4, c_3, c_2, c_1, c_0$
K dispozici máte:

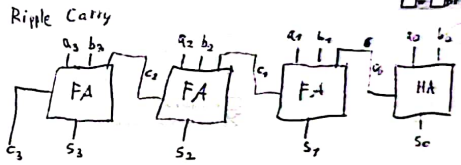
1. poloviční sčítačku (Half Adder) s jednobitovými vstupy a,b a výstupy $c_{out}s$ (dvoubitové číslo $c_{out}s$), která realizuje $c_{out}s = a + b$

2. úplnou sčítačku (Full Adder) s jednobitovými vstupy a,b,c a výstupy $c_{out}s$ (dvoubitové číslo $c_{out}s$), která realizuje $c_{out}s = a + b + c$

Dále nakreslete rychlou sčítačku Carry Lookahead adder, která použije stejné komponenty (FA, HA), ale může využít na vstupu libovolnou logickou funkci, např. $a_2 + b_2 \cdot (a_1 + a_0)$, kde operace + je logický součet or, operace · je logický součin and. Pro přehlednost si můžete nadefinovat i pomocné mezivýpočty, např. $g_0 = a_0 \cdot b_0$, $p_3 = a_3 + b_3$. Pokud neuvažujete celou sčítačku, nakreslete alespoň část.

2/5

Ripple Carry



výsledek $c = c_3, s_3, s_2, s_1, s_0$

Carry Lookahead

$p_x = a_x + b_x$
$g_x = a_x \cdot b_x$
$c_1 = (c_0 \cdot p_0) + g_0$

Skenováno pomocí CamScanner

Marie: 24/30 (total score: 24/30)

+24/1/28+

Architektura počítačů (B0B35APO) **Zkouškový test, LS2023/24**

[redacted]

← Prosíme, vyplňte zde své čestnostné identifikační číslo (uvedeno na kartě ISIC). Toto číslo bude přednostně použito pro Vaši identifikaci. Zároveň uveďte své jméno a příjmení.

Jméno a příjmení: .....

.....

Čas na vypracování testu: 90 minut. Otázky označené symbolem ♣ mohou mít více správných odpovědí. Ostatní otázky mají pouze jedna správnou odpověď.

Question 1 (+3b) Vyberte co dělá logický obvod na tomto obrázku:

3/3

- ☐ plná sčítačka (full-adder)
- ☒ poloviční sčítačka (half-adder)
- ☐ realizace registru obvodem d-latch
- ☐ multiplexor

Question 2 (±1b) Lze teoreticky sestrojit procesor pouze z hradel nand?

1/1

- ☐ NE
- ☒ ANO

Question 3 (±1b) Procesor RISC-V může pracovat s pamětí i v instrukci addi?

1/1

- ☐ ANO
- ☒ NE

Question 4 (±1b) Fáze execute přistupuje k paměti RAM?

1/1

- ☐ ANO
- ☒ NE

Question 5 (±1b) Sériová sběrnice pracující na stejné frekvenci jako paralelní sběrnice přenese méně dat?

0/1

- ☐ NE
- ☒ ANO

Question 6 (±1b) Při volání funkce v RISC-V se první argument vždy uloží na zásobník?

1/1

- ☒ NE
- ☐ ANO

Question 7
(+3b) Pokud následující 6-bitové binární číslo 110011 je kódováno v dvojkovém doplňku, kterému dekadickému číslu odpovídá?

3/3

☐ 9	☐ 22	☐ 51	☐ 19	☐ 16
☐ 26	☐ 24	☐ 18	☐ 12	☐ -45
☒ -13	☐ 35	☐ -6	☐ -1	☐ -12

+24/2/27+

Zcela jistě znáte způsob reprezentace čísel v pohyblivé řádové čárce v jednoduché přesnosti dle IEEE745. Nicméně Vaše aplikace vyžaduje, aby číslo v plovoucí řádové čárce zabíralo pouze 8 bitů. Proto jste navrhli reprezentaci, ve které má exponent 3 bity, mantisa 4 bity. Jinak chcete zachovat všechny vlastnosti IEEE754.

Question 8 (+2b) Jak se binárně zakóduje dekadické číslo 7.85 v této reprezentaci? (**Pokyny pro vyplnění:** Box zaškrtkujte pouze pokud má být daný bit nastaven na 1. Prázdný box je považován za 0. Nejméně významný bit je úplně vpravo. I pokud vychází zbytek větší než 0.5 LSB tak mantisu nahoru nezaokrouhlujte.)

2/2

☐1 ☒1 ☐1 ☒1 ☒1 ☒1 ☒1 ☒1

Question 9 (+1b) Jak bude v této reprezentaci binárně zakódováno největší možné zobrazitelné reálné číslo?

1/1

☐1 ☒1 ☒1 ☐1 ☒1 ☒1 ☒1 ☒1

Question 10 (+1b) Jaká bude dekadická hodnota tohoto největšího možného čísla? Uvažujte 3 desetinné pozice.

1/1

☐ 184.500	☐ 59.000	☐ 44.250	☐ 23.000	☐ 11.500	☐ 241.500
☒ 15.500	☐ 409.875	☐ 49.875	☐ 6.500	☐ 449.500	☐ 151.000

Question 11 (+1b) Jak bude v této reprezentaci zakódováno nejmenší nenulové kladné normalizované číslo?

1/1

☐1 ☐1 ☐1 ☒1 ☐1 ☐1 ☐1 ☐1

Question 12 (+1b) Jaká bude dekadická hodnota tohoto nejmenšího nenulového kladného normalizovaného čísla? Zaokrouhlete na 3 desetinné pozice.

1/1

☐ 0.129	☐ 0.004	☐ 0.000	☐ 0.078	☐ 0.000	☐ 0.078
☐ 0.018	☐ 0.000	☐ 0.016	☐ 0.375	☐ 0.029	☒ 0.250

Uveďte strojový kód instrukce "add" (v hexadecimálním tvaru) níže uvedeného fragmentu. Pokud některá z položek instrukce není specifikována nebo ze zadání úkolu nevyplývá, pak pro účely tohoto zkouškového testu může daná položka nabývat libovolnou hodnotu – ideálně je však tuto položku nevyplňovat. Operační kód použitých instrukcí je následující: addi:0b0010011, add:0b0110011, sw:0b0100011. Instrukce addi má fct3 = 0, instrukce add má funct7=0 a fct3 = 0, instrukce sw má fct3= 0b010.

```
addi x12, x12, -12
add x15, x15, x12
sw x15,-12(x12)
```

I	imm[12] 31:20		rs1(5), 19:15	fct3,14:12	rd(5),11:7	opcode(7),6:0
R	funct7(7) 31:25	rs2(5), 24:20	rs1(5), 19:15	fct3,14:12	rd(5),11:7	opcode(7),6:0
S	imm[11:5], 31:25	rs2(5), 24:20	rs1(5), 19:15	fct3,14:12	imm[4:0],11:7	opcode(7),6:0

Question 13 (+2b) Jak bude zakódována horní polovina instrukce "add" v hexadecimálním tvaru? Tzn. horních 16 bitů instrukce vyjádřené v strojovém kódu. (První cifru čísla vyplňte do prvního řádku, druhou cifru do druhého řádku atd. Hodnotám 10, 11, 12, 13, 14 a 15 odpovídá A, B, C, D, E a F.

2/2

☒0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☒0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☒11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☐3	☐4	☐5	☐6	☒7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15

Question 14 (+2b) Jak bude zakódována dolní polovina instrukce "add" v hexadecimálním tvaru?

2/2

☐0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☒8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☐3	☐4	☐5	☐6	☒7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15
☐0	☐1	☐2	☐3	☐4	☐5	☐6	☐7	☐8	☐9	☒10	☐11	☐12	13	☐14	☐15
☐0	☐1	☐2	☒3	☐4	☐5	☐6	☐7	☐8	☐9	☐10	☐11	☐12	☐13	☐14	☐15

Skenováno pomocí CamScanner

Question 15 (+1b) Datový typ float, který se používá v jazyce C se v paměti obvykle ukládá v:

1/1

- [ ] aditivním kódu
- [ ] inverzním kódu
- [ ] přímém kódu
- [ ] není specifikováno
- [ ] dvojkovém doplňku
- [ ] BCD kódu
- [ ] Grayově kódu
- [x] dle standardu IEEE 754
- [ ] Hammingově kódu

Question 16
(+1b) Algoritmus LRU v případě konfliktu platných řádek v cache nahrazuje

1/1

- [ ] náhodně vybraný řádek
- [ ] dosud nepoužitý řádek
- [ ] řádek s nejnižší hodnotou
- [ ] naposled přidaný řádek
- [ ] nejméně používaný řádek
- [x] nejdéle nepoužitý řádek
- [ ] naposled použitý řádek
- [ ] řádek s nejnižší adresou
- [ ] první/nejstarší přidaný řádek

Question 17
(+2b) Pokud uvažujeme dekadické číslo 27 a aditivní konstantu 15, jaká bude 6-bitová binární reprezentace takto kódovaného čísla v kódu s posunutou nulou?

0/2

- [ ] 000011
- [ ] 001110
- [ ] 000001
- [ ] 110000
- [ ] 001100
- [ ] 100110
- [ ] 001101
- [x] 101010
- [x] 111010
- [ ] 100001
- [ ] 100000
- [ ] 001011
- [ ] 010100
- [ ] 010000
- [ ] 011101

Question 18 (+5b) Nakreslete pomalou 4-bitovou sčítačku s postupným výpočtem carry (řetězová sčítačka – Ripple Carry Adder) pro součet dvou čísel $a = a_3, a_2, a_1, a_0$ a $b = b_3, b_2, b_1, b_0$ v dvojkovém doplňku. V obrázku vyznačte výsledek, číslo $c = c_4, c_3, c_2, c_1, c_0$
K dispozici máte:

1.

poloviční sčítačku (Half Adder) s jednobitovými vstupy a,b a výstupy $c_{o,s}$ (dvoubitové číslo $c_{o,s}$), která realizuje $c_{o,s} = a + b$

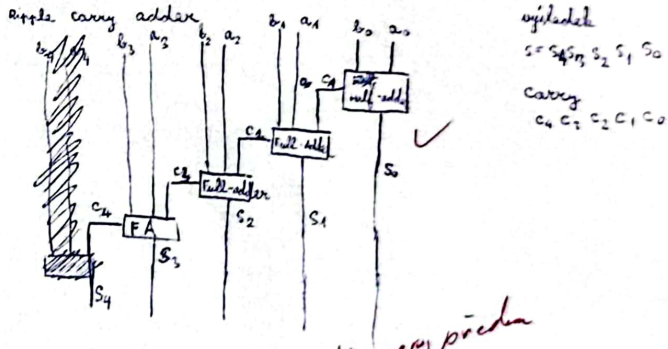
2.

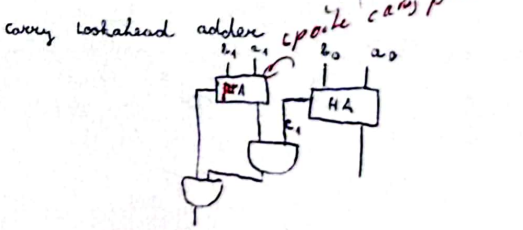
úplnou sčítačku (Full Adder) s jednobitovými vstupy a,b,c a výstupy $c_{o,s}$ (dvoubitové číslo $c_{o,s}$), která realizuje $c_{o,s} = a + b + c$

Dále nakreslete rychlou sčítačku Carry Lookahead adder, která použije stejné komponenty (FA, HA), ale může využít na vstupu libovolnou logickou funkci, např. $a_2 + b_2 \cdot (a_1 + a_0)$, kde operace + je logický součet or; operace · je logický součin and. Pro přehlednost si můžete nadefinovat i pomocné mezivýpočty, např. $g_0 = a_0 \cdot b_0$, $p_1 = a_1 + b_1$. Pokud nezvládnete celou sčítačku, nakreslete alespoň část.

- [ ] w
- [ ] p1
- [x] p2
- [ ] p3
- [ ] p4
- [ ] c

2/5





Skenováno pomocí CamScanner