Imię i nazwisko: _____ Numer indeksu: _____

Architektury systemów komputerowych

5 września 2016

czas trwania: 150–180 minut

Punkty	Ocena
90 – 100	5.0
80 – 89	4.5
70 – 79	4.0
60 – 69	3.5
50 – 59	3.0
0 – 49	2.0

Uwagi do zadań 1–2: Należy używać wyłącznie operatorów arytmetyczno-logicznych, przypisania, stałych i dodatkowych zmiennych. **Zabrania się** korzystania z instrukcji sterowania, w tym instrukcji i operatora warunkowego, oraz operatorów mnożenia, dzielenia i modulo. **Nie można** dopuścić do wystąpienia nadmiaru i niedomiaru!

Zadanie 1 (6). Przetłumacz poniższą funkcję na procedurę języka `C` o sygnaturze `unsigned num(int)`.

$$\texttt{num}\,(i) = \begin{cases} 2 \cdot i & \text{dla } i \geq 0 \\ 2 \cdot -i - 1 & \text{dla } i < 0 \end{cases}$$

Zadanie 2 (8). Przetłumacz poniższą funkcję na procedurę języka `C` o sygnaturze `int disth(int, int)`.

$$\texttt{disth}\,(x, y) = \left\lceil \frac{x - y}{2} \right\rceil$$

Zadanie 3 (8). Przetłumacz kod procedury `foobar` z asemblera x86-64 do języka C. **Należy posłużyć się** wysokopoziomowymi instrukcjami sterującymi. Używanie etykiet i instrukcji `goto` jest **niedozwolone**.

```
1  foobar:
2        xor   %rax, %rax
3  .L2:  cmp   $0, %rdx
4        jle   .L5
5        cmp   %rsi, %rax
6        jge   .L5
7        sub   (%rdi,%rax,8), %rdx
8        inc   %rax
9        jmp   .L2
10 .L5:  ret
```

Zadanie 4 (10). Przeczytaj poniższy kod w języku C i odpowiadający mu kod w asemblerze x86-64, po czym wywnioskuj wartość stałych R, S i T. W kratce poniżej należy umieścić **zwięzły** opis wnioskowania prowadzący do odpowiedzi.

```
1  long A[R][S][T];
2
3  long storeA(long i, long j, long k,
4              long value)
5  {
6    A[i][j][k] = value;
7    return sizeof(A);
8  }
```

```
1  storeA:
2      addq  %rdi, %rdx
3      leaq  (%rdx,%rsi,8), %rdx
4      leaq  (%rsi,%rsi,2), %rsi
5      salq  $5, %rdi
6      addq  %rsi, %rdx
7      addq  %rdi, %rdx
8      movq  %rcx, A(,%rdx,8)
9      movq  $1848, %rax
10     ret
```

Imię i nazwisko: _____ Numer indeksu: _____

Zadanie 5 (8). Niech `i` będzie niezerową liczbą całkowitą typu `int`, oraz `f` i `g` niezerowymi liczbami zmiennopozycyjnymi typu `float`. Podaj dowolne wartości, dla których poniższe wyrażenia będą prawdziwe:

```
          i == -i                  _____
(i & 3) != 3 && (i << 29 < 0)      _____
  (f + g) - (f + g) != 0.0         _____
(f + g) + 1.0 != f + (g + 1.0)     _____
```

Zadanie 6 (10). TLB jest zorganizowany jako czterodrożna pamięć sekcyjno-skojarzeniowa o 4 zbiorach. Wirtualna przestrzeń adresowa ma 2^{14} bajtów, a fizyczna 2^{12}. Rozmiar strony to 64 słowa. Polityka wymiany wpisów to *least recently used* – im wyższy numer znacznika LRU tym wpis jest starszy. Poniżej podano pierwotny stan TLB i 16 pierwszych wpisów tablicy stron. **Dane są w zapisie szesnastkowym!**

SET	TAG	PPN	LRU	TAG	PPN	LRU	TAG	PPN	LRU	TAG	PPN	LRU
0	07	02	2	09	0D	1	00	28	0	-	-	3
1	03	2D	0	-	-	3	-	-	3	-	-	3
2	-	-	3	-	-	3	-	-	3	-	-	3
3	1A	34	0	-	-	3	-	-	-	-	-	3

VPN	PPN	VPN	PPN
0	28	8	13
1	-	9	17
2	33	A	09
3	02	B	-
4	31	C	19
5	16	D	2D
6	-	E	11
7	-	F	0D

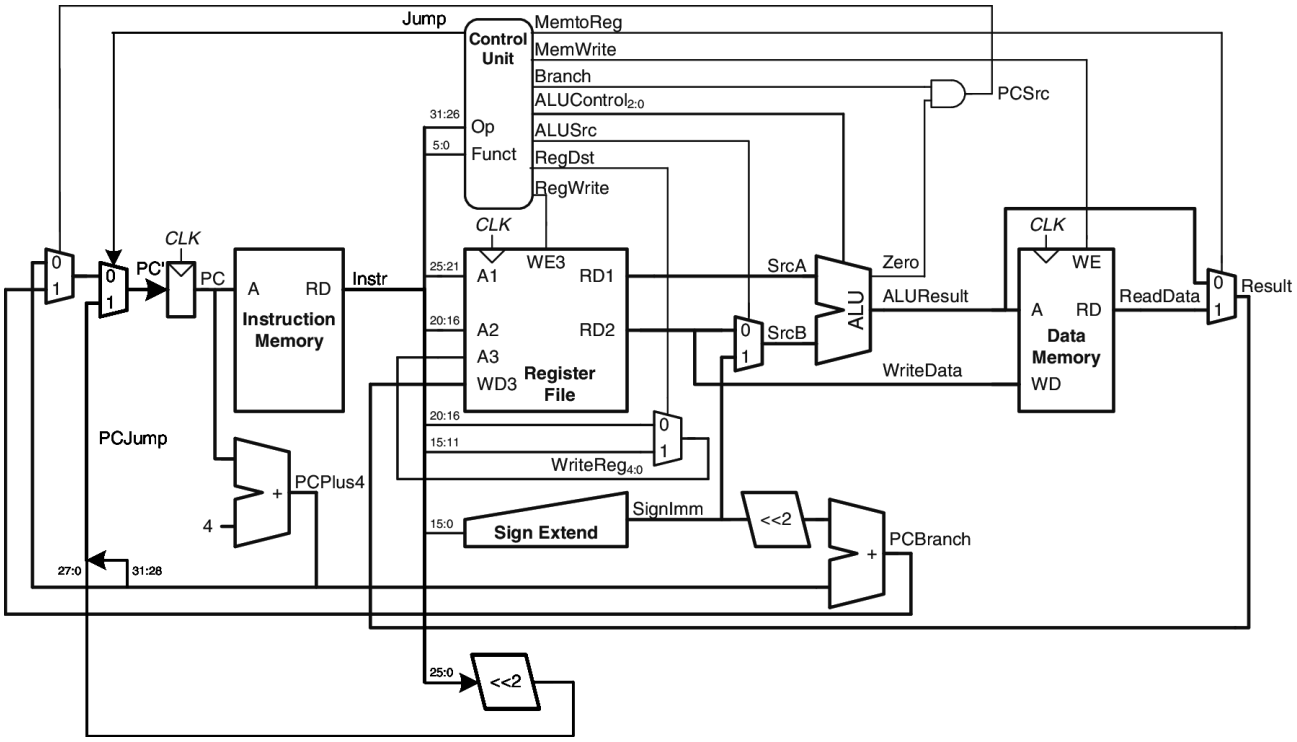
Przetłumacz adresy wirtualne podane w poniższej tabeli. Dla każdego adresu wskaż czy chybił w TLB lub wygenerował błąd strony. Podaj też ostateczny stan TLB po przetłumaczeniu wszystkich adresów. **Udzielając odpowiedzi należy użyć zapisu szesnastkowego!** Pierwszy adres został już przetłumaczony, a modyfikacja stanu TLB została uwzględniona w tabelce wyżej.

adres wirtualny	adres fizyczny	indeks TLB	znacznik TLB	chybienie w TLB	błąd strony
0024	a24	0	00	TAK	NIE
02fa					
092c					
017a					
0113					
0724					

SET	TAG	PPN	LRU	TAG	PPN	LRU	TAG	PPN	LRU	TAG	PPN	LRU
0												
1												
2												
3												

Zadanie 7 (8). Na poniższym schemacie widnieje jednocyklowa implementacja procesora MIPS. Zaproponuj kodowanie instrukcji, nowe sygnały kontrolne i modyfikacje schematu niezbędne do obsługi dodatkowej instrukcji. Podaj stan wszystkich sygnałów sterujących. Modyfikowanie ALU jest **zabronione**!

UWAGA: Na schemacie należy jedynie zakreślić modyfikowany fragment procesora i przerysować go ze zmianami do kratki!



mnemonik	typ	semantyka
movz $Rd,$Rs,$Rt	R	Reg[Rt] = 0 => Reg[Rd] := Reg[Rs]

	MemToReg	MemWrite	Branch	$\text{ALUControl}_{2:0}$	ALUSrc	RegDst	RegWrite	Jump
movz								

Imię i nazwisko: _____ Numer indeksu: _____

Zadanie 8 (10). Rozważmy potokowy procesor MIPS z implementacją obejść i obsługą hazardów, ale bez *branch delay slots*. Zakładamy, że skoki są obliczane w fazie ID, a wykonują się w etapie EX. Przyjmujemy strategię przewidywania typu „*zawsze nie wykonuj skoku*". Narysuj diagram stanu potoku wykonania poniższego kodu. Wskaż hazardy *Read-After-Write* i oznacz ścieżki przekazywania danych przy pomocy obejść. Zakładamy, że skok w linii 2 zostaje wykonany, a w linii 5 nie zostaje.

```
1       lw     $4,4($6)       IF  ID  EX  MEM  WB
2       beq    $2,$3,.L1
3       lw     $4,0($6)
4       addi   $4,$4,128
5 .L1:  bne    $2,$4,.L2
6       subu   $2,$2,$8
7 .L2:  sw     $2,28($9)
```

Uwagi do pytań otwartych: Zapewne na każde z pytań istnieje więcej niż jedna dobra odpowiedź. Zadania mają na celu sprawdzenie, czy student jest zaznajomiony z tematyką omówioną na wykładzie i właściwie posługuje się terminami związanymi z opisywanym zagadnieniem.

Zadanie 9 (8). Przedstaw proces konsolidacji pliku wykonywalnego z plików relokowalnych. Skąd system operacyjny wie pod jakie adresy załadować program i gdzie znajduje się pierwsza instrukcja programu?

Zadanie 10 (8). Podaj różnice między wyjątkiem, pułapką, a przerwaniem. Podaj przykłady zdarzeń, które je generują. Następnie opisz krótko mechanizm ich obsługi.

Zadanie 11 (8). W jakich przypadkach należy rozważyć reorganizację struktur danych w pamięci komputera zamiast obniżania asymptotycznej złożoności obliczeniowej?

Zadanie 12 (8). Z jakich przyczyn system operacyjny może sobie zażyczyć, by zawartość danych stron pamięci nie była przechowywana w pamięci podręcznej procesora?