МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Київський національний університет будівництва і архітектури

СИСТЕМНЕ ПРОГРАМУВАННЯ

Методичні вказівки до виконання курсової роботи
для підготовки здобувачів освітньо-кваліфікаційного рівня
«бакалавр» спеціальностей 122 “Комп`ютерні науки”
та 126 “Інформаційні системи і технології”

Київ 2024

УДК 681.3
С34

Укладачі: В.М. Хроленко, канд. техн. наук, доцент
В.Г. Голенков, старший викладач

Рецензент О.В. Горда, канд. техн. наук, доцент

Відповідальна за випуск Т.А. Гончаренко, канд. техн. наук, доцент

Затверджено на засіданні кафедри інформаційних технологій, протокол № 7 від 09 лютого 2024 року.

В авторській редакції

С34 Системне програмування: методичні вказівки до виконання курсової роботи / уклад.: В.М. Хроленко, В.Г. Голенков. – Київ: КНУБА, 2024. – 22 с.

Містить мету, завдання та зміст курсової роботи з дисципліни “Системне програмування”. Призначено для здобувачів першого (бакалаврського) рівня вищої освіти, які навчаються за спеціальностями 122 “Комп’ютерні науки”, 126 “Інформаційні системи і технології”.

Зміст

Загальні положення

Курсова робота з дисципліни “Системне програмування”:

- сприяє закріпленню та поглибленню здобувачами теоретичних знань з дисципліни;
- дає змогу здобувачам набути практичних навичок зі складання програм мовою Assembler, а також з розробки, тестування та налагодження програм.

Виконання курсової роботи дозволить студентам краще підготуватись до вивчення наступних спецдисциплін, передбачених навчальним планом.

Варіант курсової роботи обирається здобувачем зі списку варіантів, що наведений нижче у методичних вказівках. Номер варіанта курсової роботи відповідає номеру здобувача в списку академічної групи.

Під час виконання курсової роботи здобувач повинен:

- розібрати теоретичний матеріал за темою роботи, ознайомитись з додатковою науково-технічною літературою, що наведена у методичних вказівках, або іншими джерелами, що пов’язані з темою роботи;
- скласти алгоритм розв’язання задачі;
- скласти програму мовою Assembler;
- підготувати вихідні дані та розробити тестовий приклад для перевірки роботи програми;
- захистити роботу.

Курсова робота вважається завершеною, якщо отримано правильний результат за всіма тестовими прикладами роботи програми, оформлено пояснювальну записку згідно з вимогами, наведеними нижче, а під час захисту курсової роботи здобувач вільно орієнтується в теоретичному матеріалі та правильно відповідає на запитання, поставлені викладачем.

Оформлення пояснювальної записки до курсової роботи

Пояснювальна записка до курсової роботи має містити:

1. Опис призначення співпроцесора.

2. Постановку задачі, обмеження на характер та розмірність інформації, що обробляється, опис вихідних даних та результатів роботи.

3. Опис типів даних, що обробляються співпроцесором та були використані при написанні програми, їхні формати. Переведення значення вихідних змінних з десяткової системи числення до двійкової навести у формі, представленій у табл. 1.

Таблиця 1

Формат даних співпроцесора

Десяткове значення	Знак	Експонента	Мантиса	Тип
-247.375	1	0111	11101110110	Коротке дійсне
. . .	. . .	. . .	. . .	. . .

4. Опис порядку використання та вмісту регістрів даних при виконанні арифметичних операцій, а також операцій пересилки даних.

5. Опис вмісту регістру станів під час виконання операцій порівняння; опис наводиться у виді схематичного зображення регістрів та їх вмісту.

6. Опис основних команд співпроцесора: команд пересилання даних, арифметичних команд, команд порівняння та управління співпроцесором, які були використані при написанні програми.

7. Приклад розв’язання задачі згідно з алгоритмом, що був запропонований, з докладним аналізом кожного етапу.

8. Схему алгоритму розв’язання задачі з коментарями до неї. У звіті наводиться збільшена схема алгоритму для всієї програми та детальна схема для кожної підпрограми. Для поліпшення розуміння схеми алгоритму необхідно навести опис основних змінних програми.

9. Опис та обґрунтування тестових прикладів виконання програми.

10. Роздруківку тексту програми та результатів роботи програми.

Пояснювальна записка оформлюється на аркушах формату А4, всі аркуші нумеруються та скріплюються. Приклад оформлення титульного аркушу наведений у додатку.

Варіанти завдань до курсової роботи

Обчислити значення функції (табл. 2) та вивести на екран повідомлення:

- "Значення функції додатнє";
- "Значення функції від'ємне";
- "Значення функції дорівнює 0".

Змінні ***a, b, x, p*** – дійсні числа, значення яких подане у десятковій системі обчислення.

Таблиця 2

Варіанти завдань

№ по	***Функція***	***a***	***b***	***x***	***P***
1	$Y=$	3.456	100	10.25	-2.47
2	$Y=$	4.5	6.28	12	0.45
3	$Y = \sin^3(ax^2+bx+p)$	2.34	1.25	0.1	0.98
4	$Y= (1+ \cos(x-a))/(2b-p)$	4.5	23.56	5.23	71.3
5	$Y=\sin^2(arctg(x/a))+b*p$	2.5	10	7.125	3.45
6	$Y=(x+a)/(x^2+b)- \cos p$	2.7	3.4	12.6	0.67
7	$Y=ax^2 \sin bp$	-5.6	1.34	3.67	0.001
8	$Y=bp^3 \cos ax$	9.01	2.43	-0.005	4.67
9	$Y= \sin+ \cos bp$	3.25	0.5	0.01	1.32
10	$Y=a \sin^2(3x-b) /p$	2.35	0.78	0.035	5.89
11	$Y=-a+x^3(\cos b+ \sin p)$	0.5	0.35	7/3	9.18
12	$Y=$	2.5	10	7.125	3.45
13	$Y = \sin^3(ax^2+bx+p)$	-2.57	0.78	1.36	5.65
14	$Y= (1+ \cos(x-a))/(2b-p)$	0.125	7	2.2	3.11
15	$Y=\sin^2(arctg(x/a))+b*p$	7.12	10	7.125	-9.32
16	$Y=(x+a)/(x^2+b)- \cos p$	8.43	21.1	3	3.42
17	$Y=ax^2 \sin bp$	9.01	2.43	-0.005	4.67
18	$Y= ax \cos bp^3$	3.14	2.5	9.01	0.3
19	$Y= + tg bp$	0.5	0.35	7.3	9.18
20	$Y= (3x-b) \sin^2 p / a$	3.456	100	0.25	-2.47
21	$Y=-a \sin p +x^3\cos b$	4.5	3.47	12	0.45

Продовження табл. 2

22	*Y=*	2.5	10	7.125	3.45
23	*Y=*	6.28	2.65	1.3	10.25
24	*Y=*	0.35	0.05	37.5	1.45
25	$Y=15\ cos^2(a*p) / (x+b)$	2.17	9.11	18.21	4.65

Теоретичний матеріал за темою "Призначення та використання співпроцесора"

1. Числа з плаваючою крапкою та типи даних FPU

У процесорах Intel всі операції з плаваючою крапкою виконує спеціальний пристрій FPU (Floating Point Unit), що поставлявся спочатку у вигляді співпроцесора (8087, 80287, 80387, 80487), а починаючи з 80486DX – вбудованим в основний процесор.

Типи даних, що обробляються співпроцесором, представлені у табл. 3.

Таблиця 3

Типи даних співпроцесора

Тип даних	Кількість біт	Кількість значущих цифр	Діапазон представлення чисел
Ціле слово	16	4	-32768 … 32767
Коротке ціле	32	9	$-2*10^9$ … $2*10^9$
Довге ціле	64	18	$-9*10^{18}$ …$9*10^{18}$
Спаковане десяткове	80	18	-99..99 …+99..99 (18 цифр)
Коротке дійсне	32	7	$1.18*10^{-38}$ … $3.40*10^{38}$
Довге дійсне	64	15-16	$2.23*10^{-308}$…$1.79*10^{308}$
Розширене дійсне	80	19	$3.37*10^{-4932}$ …$1.18*10^{4932}$

Дійсні числа зберігаються у формі двійкових чисел. Двійковий запис числа з плаваючою крапкою аналогічний десятковому, тільки позиції ліворуч від коми визначаються відніманням 2 у відповідному від'ємному ступені.

Приклад 1

Перетворити число 0.625 у двійкову систему числення.

1. 0.625 – 0.5 (або 1/2)= 0.125 – результат операції додатній, його записуємо 1.

2. 0.125 – 0.25(або 1/4)= - 0.125 – результат від'ємний, записуємо 0.

3. 0.125 – 0.125(або 1/8)= 0 – записуємо 1.

У результаті перетворення отримуємо двійкове дійсне число 0.101В.

Неодмінною вимогою є запис числа у нормалізованій формі: перед десятковою крапкою має стояти одиниця. Таким чином, наше число матиме такий вигляд: $1.01В *2^{-1}$.

Приклад 2

Перетворити число 3.25 у двійкову систему обчислення.

1. Ціла частина числа 3.25 – 3. У двійковій системі обчислення вона записується як 11В.

2. Дробова частина 0.25 – 01В.

3. Загальний вигляд числа 3.25 у двійковій системі обчислення 11,01В (рис. 1).

Номер розряду	1	0	-1	-2
Значення	1	1	0	1

Рис. 1. Представлення числа у двійковій системі обчислення

Для зворотнього перетворення необхідно обчислити суму:

$1*2^1+1*2^0+0*2^{-1}+1*2^{-2}= 2+1+0.25=3.25.$

2. Формати дійсних чисел з плаваючою крапкою, використовувані в процесорах Intel

Числа з плаваючою крапкою складаються з двох частин: мантиси та порядку (експоненти) і записуються у вигляді:

$$N = M \cdot S^p,$$

де: N - число з плаваючою крапкою;

M - мантиса числа;

S - основа системи обчислення;

p - порядок числа.

Приклад 3

- $5.729 = 0.5729 \cdot 10^1$, де 0.5729 – мантиса, 10 – основа системи обчислення, 1 – порядок числа;
- $1.101 * 2^{-1}$, де 1.101 – мантиса двійкового числа, -1 – порядок.

Для того, щоб не зберігати від'ємний порядок числа, він подається у так званому зміщеному вигляді, тобто збільшеним на певну константу Т: p' = p + T. Наприклад, для короткого дійсного на $T = 2^6$ =64, довгого дійсного – 1023, розширеного дійсного – 16 383. Тобто для короткого дійсного замість $-64 \le p \le 63$, значення порядку буде $0 \le p' \le 127$, де $p' < 64$ – від'ємний порядок, $p' \ge 64$ – додатній порядок. Це спрощує дії над числами та полегшує виконання операцій порівняння.

Крім того, число треба записувати у нормалізованій формі: перед крапкою має бути одна значуща цифра (див. приклад 1).

Розміри порядку (експоненти) та мантиси числа наведені у табл. 2, а формат запису двійкового числа – на рис. 2, де М – кількість розрядів під мантису (відповідно до типу числа); Р – кількість розрядів під експоненту; 1 розряд під знак числа (1 – від'ємне, 0 – додатнє).

Таблиця 4

Формати чисел з плаваючою крапкою

Тип даних	Кількість біт			Кількість значущих цифр
	Загальна	Мантиса	Порядок (експонента)	
Коротке дійсне	32	23	8	7
Довге дійсне	64	52	11	15 -16
Розширене дійсне	80	64	15	19

Такий формат представлення даних відповідає міжнародному стандарту IEEE854. Його перевагою, зокрема, є те, що операції порівняння

Знак	Експонента										Мантиса									
								0			1									

Рис. 2. Формат двійкового числа з плаваючою крапкою

дійсних чисел відбуваються за тим самим алгоритмом, що і порівняння цілих чисел.

FPU виконує всі обчислення у 80-бітному розширеному форматі, а 32- і 64-бітні числа використовуються для обміну даними з основним процесором та пам'яттю.

У 32- і 64-бітному форматах, на відміну від 80-бітного, одиниця ліворуч від коми не зберігається.

Приклад 4

Десяткове значення -247.375 у двійковій системі обчислення має вигляд:

Знак	
1	11110111,011B

У нормалізованій формі:

Знак	
1	$1{,}1110111011B*2^7$

Таблиця 5

Формат дійсного числа у двійковій системі числення

Знак	Порядок	Мантиса	Тип	Примітка
1	10000110	11101110110…0	Коротке дійсне	Порядок збільшений на 127
1	1000000110	11101110110…0	Довге дійсне	Порядок збільшений на 1023
1	100000000000110	111101110110…0	Розширене дійсне	Порядок збільшений на 16383

Крім звичайних чисел, формат FPU передбачає декілька спеціальних типів даних, що можуть утворюватися в результаті математичних операцій і над якими можна виконувати наступні операції:

- *Позитивний нуль:* усі біти числа встановлені в нуль;

- *Негативний нуль:* знаковий біт – 1, всі інші біти – нулі;

- *Позитивна нескінченість:* знаковий біт – 0, усі біти мантиси – 0, усі біти експоненти – 1;

- *Негативна нескінченість:* знаковий біт – 1, усі біти мантиси – 0, усі біти експоненти – 1;

- *Денормалізоване число:* усі біти експоненти – 0 (використовуються для роботи з дуже малими числами);

- *Невизначеність:* знаковий біт – 1, перший біт мантиси (перші два для 80-бітних чисел) – 1, а інші - 0 , усі біти експоненти – 1;

- *Нечисло типу SNAN (сигнальне):* усі біти експоненти – 1, перший біт мантиси – 0 (для 80-бітних чисел перших два бита мантиси -10), а серед інших бітів є одиниці;

- *Нечисло типу QNAN (тихе):* усі біти експоненти – 1, перший біт мантиси (перші два для 80-бітних чисел) – 1, а серед інших є одиниця. Невизначеність – один із варіантів QNAN;

- *Невизначене число:* всі інші ситуації.

Інші формати даних FPU також припускають невизначеність – одиниця в старшому біті і нулі в інших для цілих чисел; старші 16 біт – одиниці для спакованих десяткових чисел.

3. Архітектура співпроцесора

FPU надає вісім регістрів для збереження даних (рис. 3) і п'ять допоміжних регістрів (рис. 4).

Регістри даних FPU R0-R7 – це 80 – розрядні регістри. Їх робота організована за принципом стеку LIFO. Доступ до кожного регістра здійснюється за логічним ім’ям ST(0) – ST(7). Вершиною стеку вважається регістр ST(0).

Коли стек порожній, покажчик вершини стеку TOP = 7 і регістр R7 має логічне ім’я ST(0). Коли у стек записується число, покажчик вершини стеку TOP зменшується на 1, вершина стеку переміщується вище. Стек має кільцеву організацію. Коли він заповнюється, наступне значення записується поверх того, що було записане першим. При цьому буде згенеровано особливі ситуації: порушення стеку та неіснуюча операція (див. регістр станів).

R0				ST(3)		
R1				ST(4)		
R2				ST(5)		
R3				ST(6)		
R4				ST(7)		
R5				ST(0)	⇐	TOP=5
R6				ST(1)		
R7				ST(2)		
	79		0			

Рис. 3. Регістри даних FPU

SR								FIP
CR								FDP
TW					47		0	
	15		0					

Рис. 4. Регістри управління FPU

Регістр станів SR містить слово стану FPU (рис. 5):

B – зайнятість FPU;

C3 – прапорець умови 3;

TOP – покажчик вершини стеку регістрів даних;

C2 – прапорець умови 2;

C1 – прапорець умови 1;

C0 – прапорець умови 0;

ES – загальний прапорець помилки, дорівнює 1, якщо виникла особлива ситуація.

SF – помилка стеку.

PE – прапорець неточного результату.

UE – прапорець антипереповнення;

OE – прапорець переповнення;

ZE – прапорець ділення на 0;

DE – прапорець денормалізованого операнда;

IE – прапорець неприпустимої операції.

15	14	13	12	11	10	9	8
B	C3	TOP			C2	C1	C0
7	6	5	4	3	2	1	0
ES	SF	PE	UE	OE	ZE	DE	IE

Рис 5. Регістр станів SR

Біти C3-C0 відповідають бітам регістра прапорців основного процесора (C0– CF, C1 – не має, C2 – PF, C3 - ZF) та використовуються при виконанні арифметичних операцій, операцій порівняння та умовних переходів.

Регістр управління CR, біти якого визначають спосіб округлення результатів, точність результатів деяких арифметичних команд, а також маскують відповідні виняткові ситуації. Вміст регістру може змінюватися програмно.

Регістр тегів TW містить вісім пар бітів, що описують зміст кожного регістра даних. Біти 15 –14 описують регістр R7, 13—12 – R6 тощо. Якщо пара біт дорівнює 11, відповідний регістр порожній, 00 – регістр містить число, 01 – нуль,10 – нечисло, нескінченність, денормалізоване число, непідтримуване число.

Регістри FIP і FDP містять відповідну адресу останньої виконаної команди і адресу її операндів.

4. Команди FPU

4.1 Особливості формування команд

Бінарні операції, наприклад, додавання, припускають декілька форм.

- Вказані два операнди:

< команда > <операнд1>,<операнд2>.

Тут <операнд1>, <операнд2> - регістр даних FPU. При цьому один з них має бути ST(0), а другим - ST(i).

Після виконання операції покажчик стеку збільшується на 1, результат записується до першого операнда, другий операнд зі стеку видаляється.

- Вказаний один операнд:

<команда > <операнд >.

<Операнд > - регістр даних, змінна у пам’яті, чисельна константа.

Операція виконується над вершиною стеку та вказаним операндом. Результат записується до вершини замість попереднього значення. Покажчик стеку не змінюється.

- Команда вказується без операндів.

Операція здійснюється над регістрами ST(0) та ST(1). Вміст ST(0) виштовхується зі стеку, покажчик стеку зменшується на 1. Результат записується до нової вершини стеку – колишнього ST(1) (рис. 6).

До виконання операції				Після виконання операції	
ST(7)				. . .	
ST(0)				ST(7)	
ST(1)				ST(0)	Результат

Рис. 6. Використання регістрів при виконанні унарних операцій

Унарні операції виконуються над операндом або над операндом та вершиною стеку. Запис результату залежить від реалізації.

4.2 Команди пересилки даних FPU

Команди пересилки даних FPU унарні та мають два напрямки: завантаження до стеку та зчитування (копіювання) зі стеку. Вони еквівалентні командам завантаження до стеку PUSH та виштовхування зі стеку POP. При завантаженні виконується:

- зменшення покажчика стеку TOP на 1, таким чином створюється нова вершина стеку;
- передача операнда в нову вершину стеку.

При зчитуванні (виштовхуванні) зі стеку виконується:

- передача значення ST(0) до операнда;
- звільнення регістра ST(0), збільшення покажчика стеку TOP на 1.

При копіюванні зі стеку значення ST(0) передаються до операнда, вершина стеку не змінюється.

Команди, що вказані у таблиці 3, мають по одному операнду.

В залежності від напрямку операції операндами можуть бути:

- запис у стек – імена змінних, константи, ST(n);

- зчитування (копіювання) до змінної або ST(n).

Таблиця 6

Команди пересилки даних

	Завантажити до стеку	Скопіювати у приймач	Виштовхнути зі стеку у приймач
Дійсне число	FLD	FST	FSTP
Ціле	FILD	FIST	FISTP
Спаковане десяткове	FBLD	-	FBSTP

Приклад 5

FLD Variable; завантажити довге ціле

FLD ST(2); завантажити значення з ST(2) до вершини стеку

FILD M; завантажити цілочисельне M

FST ST(5); скопіювати ST(0) до регістра стеку FST R; скопіювати ST(0) до змінної R

Команда: ***FCMOVcc*** приймач, джерело

Призначення: умовна пересилка даних

В залежності від виконання (або невиконання) певної умови копіює вміст джерела (регістр ST(n)) у приймач (тільки ST(0)), якщо виконується відповідна умова.

Таблиця 7

Команди умовної пересилки даних

Команда	Значення прапорців	Дія після FCOM
FCMOVE	ZF=1	Якщо дорівнює
FCMOVNE	ZF=0	Якщо не дорівнює
FCMOVB	CF=1	Якщо менше
FCMOVBE	CF=1 ZF=1	Якщо менше або дорівнює
FCMOVNB	CF=0	Якщо не менше
FCMOVNBE	CF=0 ZF=0	Якщо не менше або дорівнює
FCMOVU	PF=1	Якщо не порівнянні
FCMOVNU	PF=0	Якщо порівнянні

Команда: ***FXCH*** джерело

Призначення: поміняти місцями два регістри стеку

Обмін місцями вмісту регістра ST(0) і джерела (регістр ST(n)). Якщо операнд не зазначений, обмінюється вміст ST(0) і ST(1).

4.3 *Арифметичні операції*

Таблиця 8

Команди арифметичних операцій

	Дійсне число	Дійсне з вишт. зі стеку	Ціле число
Додавання	FADD	FSDDP	FIADD
Віднімання	FSUB	FSUBP	FISUB
Добуток	FMUL	FMULP	FIMULP
Ділення	FDIV	FDIVP	FIDIV

Кожна команда має два операнда. Результат виконання операції зберігається у першому операнді.

Арифметичні команди з виштовхуванням зі стеку звільнюють ST(0) та збільшують TOP на 1.

Команда: ***FABS***

Призначення: знайти абсолютне значення ST(0)

Якщо ST(0) був від'ємним числом, переводить його в додатне.

Команда: ***FCHS***

Призначення: змінити знак ST(0)

Змінити знак ST(0), перетворює від'ємне число в додатне і навпаки.

Команда: ***FRNDINT***

Призначення: округлити до цілого ST(0)

Округлити значення ST(0) до цілого числа відповідно до режиму округлення, заданим бітами RC.

Команда: ***FSQRT***

Призначення: добути квадратний корінь

Приклад 6

FADD ST(i), ST; ST(i) ← ST(i) + ST

FADD ST(i); ST ← ST+ ST(i)

`FADD D; ST ← ST+ D, D – дійсна змінна у пам'яті`

4.4 *Операції порівняння*

Таблиця 9

Операції порівняння

Унарні операції			Без операндів
	Порівняння	Порівняння з виштовх. зі стеку	Порівняння з виштовх. двох чисел зі стеку
Дійсне число	FCOM	FCOMP	FCOMPP
Дійсне без врахування порядку	FUCOM	FUCOMP	FUCOMPP
Ціле число	FICOM	FICOMP	FICOMPP
Ціле з устан. прапорців EFLAGS	FCOMI	FCOMIP	
Ціле без врахув. порядку	FUCOMI	FUCOMIP	

Команди виконують порівняння вмісту регістра ST(0) зі вказаним операндом (32- або 64-бітна змінна або регістр ST(n), якщо операнд не визначений – ST(1)), і встановлюються прапорці C0, C2 і C3 відповідно до таблиці 7.

Таблиця 10

Прапорці порівняння FPU

Умова	C3	C2	C0
ST(0)>джерело	0	0	0
ST(0)<джерело	0	0	1
ST(0)=джерело	1	0	0
Непорівняні	1	1	1

Якщо один з операндів – нечисло, або число, що не підтримується, виникає особлива ситуація «неприпустима операція».

Після виконання команд FSTSW AX (завантажити регістр SR в AX) і SAHF (переслати AH в FLAGS) значення C3, C2 і C0 записуються у відповідні біти регістра прапорців ZF, PF і CF, після чого всі умовні команди (Icc, CMOVcc, SETcc) можуть використовувати результат порівняння, як після команди CMD.

Команда FCOMP після виконання порівняння виштовхує зі стеку ST(0) (помічає його як порожній і збільшує TOP на 1), а команда FCOMPP виштовхує зі стеку і ST(0), і ST(1).

Команда: ***FTST***

Призначення: перевірити, чи не містить SP(0) нуль

Порівнює вміст ST(0) з нулем і виставляє прапорці C3, C2 і C0 аналогічно іншим командам порівняння.

4.5 *Трансцендентні операції*

Команда: ***FSIN***

Обчислює синус числа в ST(0), зберігає результат в тому ж самому регістрі. Операнд вважається заданим в радіанах.

Команда: ***FCOS***

Обчислює косинус числа в ST(0), зберігає результат в тому ж самому регістрі. Операнд вважається заданим в радіанах.

Команда: ***FSINCOS***

Обчислює синус та косинус числа в ST(0), зберігає результат у стеку та зменшує покажчик стеку на 1. Операнд вважається заданим в радіанах.

Команда: ***FPTAN***

Обчислює тангенс числа в ST(0), зберігає результат в тому ж самому регістрі. Потім дописує в стек 1 так, що ST(0) містить 1, а ST(1) – результат операції. Операнд вважається заданим в радіанах.

Додаток

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Кафедра інформаційних технологій

КУРСОВА РОБОТА

з дисципліни “Системне програмування”

на тему: “Призначення та використання співпроцесора”

Студента (ки) _____ курсу _____ групи
напряму підготовки_____
спеціальності_____

_____
(прізвище та ініціали)

Керівник_____
(посада, прізвище та ініціали)

Національна шкала _____
Кількість балів: _____Оцінка: ECTS ____

Члени комісії

_____ _____
(підпис) (прізвище та ініціали)

Список літератури

1. Системне програмування: методичні вказівки до виконання лабораторних робіт / уклад.: В.М. Хроленко, В.Г. Голенков. – Київ: КНУБА, 2022. – 20 с.

2. Функції та принципи роботи математичного співпроцесора. URL: https://ukrbukva.net/

3. Співпроцесори. Призначення, система команд на прикладі процесорів Intel x86. - реферати та учбові матеріали на um.co.ua

Навчально-методичне видання

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Київський національний університет будівництва та архітектури

СИСТЕМНЕ ПРОГРАМУВАННЯ

Методичні вказівки

до виконання курсової роботи

для здобувачів спеціальностей 122 “Комп`ютерні науки”

та 126 “Інформаційні системи і технології”

Комп’ютерне верстання

Підписано до друку 22.02.2024 Формат 60 × 84 1/ 16
Ум. друк. арк. 1,16. Обл.-вид. арк. 1,25.
Електронний документ. Вид № 59/ІІІ-17.

Видавець і виготовлювач
Київський національний університет будівництва і архітектури

Повітрофлотський проспект, 31, Київ, Україна, 03037
Свідоцтво про внесення до Державного реєстру суб’єктів
Видавничої справи ДК №808 від 13.02.20