Лекція 9.

Тема. Канали передачі та прийому інформації.

Розділ. Практичні реалізації деяких елементів протоколів обміну інформацією.

План лекції:

9.1. Використання поліноміальної арифметики в каналах передачі / прийому інформації.

9.1.1. Циклічне кодування.

9.1.2. Алгоритм підрахунку CRC16 контрольних сум байт-пакетів

9.2. Скремблери-дескремблери.

9.3. Використання стандартного сервісу СМС повідомлень для обміну інформацією.

9.1. **Використання поліноміальної арифметики в каналах передачі / прийому інформації.**

Поліноміальна арифметика виконує арифметичні дії над поліномами. Поліном – це лінійна комбінація добутків цілих степенів заданого набору змінних з постійними коефієнтами, наприклад:

$u(x) = u_n \bullet x_n + u_{n-1} \bullet x_{n-1} \bullet ..... \bullet u_1 \bullet x_1 + u_0 \bullet x_0$, де

$u_n,...,u_0$ коефіцієнти, що складають певну систему S.

x - змінна полінома, формальна змінна без визначеного значення. Система S – це множина цілих чисел в діапазоні від 0 до $m-1$, між якими можуть бути виконані арифметичні дії, а саме додавання, віднімання, множення, ділення по модулю m. В системах передачі інформації використовується окремий випадок цієї арифметики – поліноміальна арифметика по модулю 2, в якому кожний коефіцієнт дорівнює 0 або 1. Наприклад, для x=2: $u(x) = 1 \bullet 2^0 + 0 \bullet 2^1 + 1 \bullet 2^2 + 1 \bullet 2^3$. Всі дії з поліномами виконуються над коефіцієнтами, значення x не грає ніякої ролі. Два полінома вважаються рівними при однакових коефіцієнтах навіть, якщо значення x для них різне (наприклад, в одному x=10 а в іншому x=2). Тому поліном звичайно записують так: $u(x) = 1 \bullet x^0 + 0 \bullet x^1 + 1 \bullet x^2 + 1 \bullet x^3$, до того ж доданок з нульовими коефіцієнтами не вписують, одиничні коефіцієнти вказують уявно і доданки записують по мірі змешьшення степені зліва направо, тобто $u(x) = x^3 + x^2 + 1$. Головна віддмінність арифметичних операцій над многочленами є в тому, що між коефіцієнтами поліному немає ніякого зв'язку, тому поняття переносу в поліноміальній арифметиці відсутнє. Біт переносу або запозичення просто не враховується. Таким арифметичним операціям відповідає операція $\oplus$ «Виключне АБО». Поліноміальна арифметика використовується в циклічному кодуванні та підрухунку контрольних сум.

9.1.1. Використання циклічного кодування для реалізації процесу передачі / прийому інформації.

Алгоритм циклічного кодування широко використовується в реалізації процессу обміну в каналах передачі / прийому інформаціїї.

Циклічний код – це лінійний цифровий код, що володіє властивістю циклічності, тобто кожна циклічна перестановка кодового слова також є кодовим словом. Якщо символьну послідовність $Y = a_0 a_1 a_2 ... a_{n-1}$ записати у вигляді полінома $Y(x) = a_0 + a_1 x + a_2 x^2 + ... + a_{n-1} x^{n-1}$, де x-фіктивна змінна, то серед всьої сукупності можливих значень полінома Y(x), можна відібрати ті, які володіють додатковою властивістю циклічності, тобто кожен слідуючий код отримується циклічним зсувом попереднього на один розряд (справа наліво циклічно). Зсув кода справа наліво на один розряд для Y(x) рівноцінно множенню Y(x) на x. Такий набір кодів називається циклічним набором кодів, або просто циклічним кодом.

Циклічний код будується на основі породжуючого полінома $g(x)$ степеня r=n-k, де n-1 - степінь (розрядність) циклічного кода, k-кількість циклічних кодів.

$g(x)$ - породжуючий поліном циклічного коду є дільником двочлена $x^n - 1$. Всі коди циклічного коду можна записати у вигляді матриці

$$\begin{bmatrix} g(x) \\ Xg(x) \\ ... \\ x^{k-1} g(x) \end{bmatrix}$$

Наприклад, для породжуючого полінома $g(x) = x^3 + x + 1$, n=7 породжуюча матриця має вигляд

$$G = \begin{bmatrix} 001011 \\ 010110 \\ 101100 \\ 011001 \\ 110010 \\ 100101 \end{bmatrix}$$

Із властивостей циклічних кодів випливає, що будь який циклічний код з повного набору ділиться на $g(x)$ без залишку. Тоді виходить, що будь який набір $Y(x) = a_0 + a_1 x + a_2 x^2 + ... + a_{n-1} x^{n-1}$, який не описується породжуючою матрицею не ділиться на $g(x)$ без залишку. Це й надає можливість використовувати циклічні коди для виявлення помилок при передачі кодів. Нехай $M = m_0 m_1 m_2 ... m_{k-1}$ інформаційна послідовність, її також можна представити поліномом, G - породжуюча матриця, тоді над послідовністю M можна виконати операцію перетворення (кодування):

$$MG = (m_0 m_1 ... m_{k-1}) \begin{bmatrix} g(x) \\ xg(x) \\ ... \\ x^{k-1} g(x) \end{bmatrix} = m(x)g(x)$$

та використати послідовність $MG = m(x)g(x)$ замість $M = m_0 m_1 m_2 ... m_{k-1}$ в каналі зв'язку з можливістю виявлення та/або виправлення деяких помилок.

Приклад. $g(x) = x^8 + x^7 + x^6 + x^4 + 1$.

Нехай інформаційний код M=1000111 записаний від молодшого розряду до старшого зліва направо, тобто $m(x) = x^6 + x^5 + x^4 + 1$. Тоді $m(x)g(x) = (x^8 + x^7 + x^6 + x^4 + 1)(x^6 + x^5 + x^4 + 1)$. Отримаємо добуток:

$$Y = x^8 + x^7 + x^6 + 0 + x^4 + 0 + 0 + 0 + 1$$

$$g(x) = x^6 + x^5 + x^4 + 0 + 0 + 0 + 1$$

$$Y * 1 = \overline{x^8 + x^7 + x^6 + 0 + x^4 + 0 + 0 + 0 + 1}$$

$$\oplus$$

$$Y * x^4 = x^{12} + x^{11} + x^{10} + 0 + x^8 + 0 + 0 + 0 + x^4$$

$$\overline{x^{12} + x^{11} + x^{10} + 0 + 0 + X^7 + X^6 + 0 + 0 + 0 + 0 + 0 + 1}$$

$$\oplus$$

$$Y * x^5 = x^{13} + x^{12} + x^{11} + 0 + x^9 + 0 + 0 + 0 + x^5$$

$$\overline{x^{13} + 0 + 0 + x^{10} + x^9 + 0 + x^7 + x^6 + x^5 + 0 + 0 + 0 + 0 + 1}$$

$$\oplus$$

$$Y * x^6 = x^{14} + x^{13} + x^{12} + 0 + x^{10} + 0 + 0 + 0 + x^6$$

$$\overline{x^{14} + x^{12} + x^9 + x^7 + x^5 + 0 + 0 + 0 + 0 + 1}$$

Або у вигляді кодової символьної послідовності Y=100001010100101. Отримана послідовність передається в канал зв'язку замість інформаційного коду.

9.1.2. Алгоритм підрахунку CRC16 контрольних сум байт-пакетів.

CRC (Cyclic Redundancy Code - циклічний надлишковий код) . Алгоритм розрахунку контрольної суми для повідомлення, що передається, заснований на поліноміальною арифметиці.

Головна ідея алгоритму CRC полягає у поданні повідомлення у вигляді величезного двійкового числа (послідовності), діленні його на інше фіксоване двійкове число і використанні залишку від цього поділу в якості контрольної суми. Отримавши повідомлення, приймач повинен виконати аналогічну дію і порівняти отриманий результат з прийнятою контрольною сумою. Повідомлення вважається достовірним, якщо виконується ця рівність.

Алгоритм CRC базується на полиноміальній арифметиці по модулю 2, а це означає, що повідомлення, дільник і залишок можуть бути представлені у вигляді поліномів з двійковими коефіцієнтами або у вигляді рядків бітів,

кожен з яких є коефіцієнтом полінома. Дії, що виконуються під час обчислення CRC, є арифметичними операціями без урахування біту перенесення. Тобто додавання і віднімання виконується побітово без урахування біту перенесення, завдяки чому ці дві операції дають еквівалентний результат. Операції додавання і віднімання в цьому випадку ідентичні операції XOR («Виключне АБО»).

Ділення виконується за аналогією зі звичайним арифметичним поділом стовпчиком з тією відмінністю, що замість віднімання дільника від діленого використовується операція XOR. У програмній реалізації цього алгоритму замість дільника зсувається ділене. Зсув здійснюється вліво по одному біту. При цьому виконується перевірка зсунуиого біту: якщо він дорівнює одиниці, виконується операція XOR дільника (полінома) зі старшими розрядами діленого (повідомлення).

Щоб виконати обчислення CRC, необхідно вибрати дільник - поліном. Важливою характеристикою, що визначає подальші розрахунки, є степінь полінома або його ширина W (від англійського Ширина - ширина). Зазвичай вибирається степінь 16 або 32, так як вони є кратними розрядності регістрів сучасних процесорів, що значно спрощує реалізацію алгоритмів CRC.

Степінь полінома - дійсна позиція старшого біта. Позиції бітів відлічуються, починаючи з нульової. Вибравши поліном, в кінець повідомлення додаються W нульових бітів.

Стандартний поліном CRC-16 має значення "8005" у шістнадцятиковому представленні або 1000 0000 0000 0101 в двійковому представленні (як правило, найстарша одиниця не враховується при розрахунках).

При розрахунках CRC використовується абстрактний регістр CRC з розрядністю, рівною ширині полінома W, який зберігає поточне значення обчислюваної контрольної суми. Крім степеня полінома вибирають початкове значення регістра CRC і значення, яке комбінується через XOR з остаточним вмістом регістра. Кращим вибором для ініціалізації регістра є значення FFFFh (або 1111 1111 1111 1111 в двійковому форматі), яке забезпечить можливість виявити нульові байти. Відповідно, остаточне значення регістра буде комбінуватися по XOR також з FFFFh.

Простий алгоритм розрахунку CRC виконується наступним чином:

1. У регістр CRC заноситься початкове значення FFFFh.
2. В кінець повідомлення додається W нульових бітів.
3. Вміст регістра зсувається вліво на 1 біт, і в останню (нульову) позицію заноситься черговий, ще не оброблений біт даних.
4. Якщо з регістра був висунутий біт зі значенням «1», то вміст регістра комбінується з XOR з поліномом. Якщо значення біта одно "0", XOR не виконується.
5. Кроки 3 і 4 виконуються, поки не будуть оброблені всі дані.
6. Остаточний вміст регістра комбінується з XOR із значенням FFFFh.

9.2. Скремблери-дескремблери.

Суть скремблювання полягає в побітній зміні потоку даних, що проходить через систему. Практично єдиною операцією, що використовується в скремблері є XOR - "побітне виключне АБО». Паралельно проходженню інформаційного потоку в скремблері за певним правилом синхронно генерується кодуючий бітовий потік. Як пряме, так і зворотне шифрування здійснюється виконанням операції XOR між кодуючим потоком і вихідним / вихідним.

Генерація кодуючої послідовності біт проводиться циклічно від невеликого початкового коду - ключа за наступним алгоритмом. З поточного набору біт вибираються значення певних розрядів і складаються по XOR між собою. Всі розряди зсуваються на 1 біт, а тільки що отримане значення ("0" або "1") поміщається в молодший розряд, що звільнився. Значення, що знаходилося в старшому розряді до зсуву, додається в кодуючу послідовність, стаючи черговим її бітом (рис.9.1.)

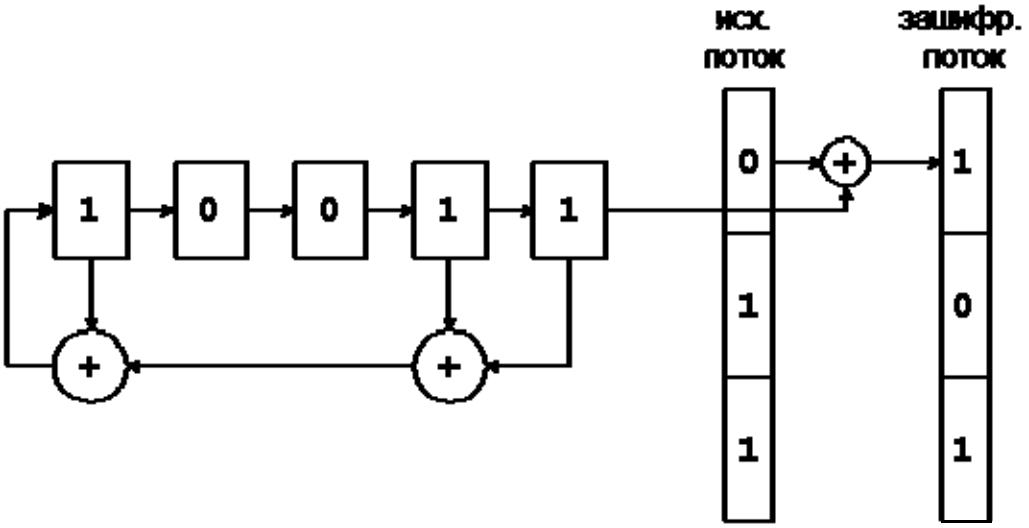

Рисунок 9.1. Структурна схема скремблера.

9.3. Використання стандартного сервісу SMS **повідомлень для обміну інформацією.**

Існуюча система SMS повідомлень мобільного зв’язку дає можливість використання її для обміну керуючими або інформаційними повідомленнями в системах предачі / прийому інформації.

Специфікація SMS визначає шлях для відправки SMS повідомлень через мобільний телефон або GSM / GPRS модем. Більшість бездротових модемів та мобільних телефонів можна використовувати для відправки та приймання SMS повідомлень. Для відправки SMS-повідомлення необхідно використати SIM-карту оператора мобільного зв’язку та підключити модем до компьютера. Після підключення GSM / GPRS модема до комп'ютера, можна керувати GSM / GPRS модемом шляхом відправки інструкцій до нього. Інструкції, що використовуються для управління GSM / GPRS модемом називаються AT командами. Dialup-модеми, мобільні телефони і GSM / GPRS модеми

обов’язково підтримують стандартний набір AT команд. На додаток до цього стандартного набору AT команд, сучасні GSM / GPRS модеми підтримують розширений набір AT команд. Розширені AT команди для відправки та отримання повідомлень SMS перераховані в таблиці 9.1.

Таблиця 9.1. AT команди для відправки та отримання повідомлень SMS.

AT команди	**Розшифровка**
+CMGS	Send message
+CMSS	Send message from storage
+CMGW	Write message to memory
+CMGD	Delete message
+CMGC	Send command
+CMMS	More messages to send

Один із способів перевірки роботи AT команд з GSM / GPRS модемом полягає у використанні термінальної програми. Функція термінальної програми такий: Вона посилає символи, які ви набрали на клавіатурі, на GSM / GPRS модем. Потім вона відображає відповідь, яку отримує від GSM / GPRS модема, на екрані. Термінальна програма для Microsoft Windows має назву HyperTerminal.

Нижче наведено простий приклад, який демонструє, як за допомогою AT команд і HyperTerminal програми Microsoft Windows відправити повідомлення SMS. Лінії командного рядка, які повинні бути введені в HyperTerminal виділені жирним шрифтом. Решта рядків є відповіддю GSM / GPRS модема.

AT
OK
AT + CMGF = 1
OK
AT + CMGW = "+380661234567"
> Проста демонстрація текстових повідомлень SMS.
+ CMGW: 1
OK
AT + CMSS = 1
+ CMSS: 20
OK

Пояснення того, що зроблено в наведеному вище прикладі:

Рядок 1: "AT" відправляється в GSM / GPRS модем, щоб перевірити з'єднання. GSM / GPRS модем посилає назад код результату "OK" (лінія 2),

який означає, що зв'язок між HyperTerminal програмою і GSM / GPRS модемом працює нормально.

Рядок 3: AT команда + CMGF використовується для вказівки GSM / GPRS модему працювати в текстовому режимі SMS. У відповідь код "OK" (рядок 4) вказує що "AT + CMGF = 1" був успішно виконаний. Якщо в результаті повертається код "ERROR", цілком ймовірно, що GSM / GPRS модем не підтримує режим SMS повідомлень. Для перевірки треба ввести "AT + CMGF =?" У програмі HyperTerminal. Якщо відповідь "+ CMGF: (0,1)" (0 = PDU режим і 1 = текстовий режим), то режим SMS повідомлень підтримується. Якщо відповідь "+ CMGF: (0)", то режим SMS повідомлень не підтримується.

Рядки 5 і 6: AT + команди CMGW використовуються для відправлення повідомлення SMS «Проста демонстрація текстових повідомлень SMS.». "**+380661234567**" є номер мобільного телефону отримувача. Після введення номера мобільного телефону, необхідно натиснути кнопку Enter на клавіатурі. GSM / GPRS модем буде повертати рядки з початковим символом ">", після якого можна вводити текст SMS-повідомлення " Проста демонстрація текстових повідомлень SMS.». По закінченні необхідно натиснути Ctrl + Z на клавіатурі.

Рядок 7: "+ CMGW: 1" говорить, що текстовому SMS повідомленню присвоюється індекс 1. Це вказує на розташування текстових повідомлень SMS в пам’яті модема.

Рядок 9: код результату "OK" вказує на успішне виконання команди AT + CMGW.

Рядок 10: AT + CMSS команда використовується для відправки повідомлення SMS 1 з пам’яті GSM / GPRS модема.

Рядок 11: "+ CMSS: 20" говорить про ідентифікаційний номер відправленого текстового повідомлення SMS.

Рядок 13: Код результату "OK" показує виконання команди AT + CMSS успішно.

Для реалізації відправки SMS повідомлень з додатків, необхідно написати код на C, C + +, Java, Visual Basic, Delphi або інших мовах програмування для підключення модему і відправки AT команд на GSM / GPRS модем. Або використати SMS Messaging API (інтерфейс прикладного програмування).