অধ্যায়-২

বাইনারি অ্যারিথমেটিক অপারেশন
(Understand the Binary Arithmetic Operation)

২.১ দশমিক, বাইনারি, অক্টাল ও হেক্সাডেসিমাল সংখ্যা পদ্ধতির সংজ্ঞা (Define decimal, binary, octal and hexadecimal number system) ঃ

সংখ্যা পদ্ধতি ঃ যে পদ্ধতিতে কোনো সংখ্যা লেখা বা প্রকাশ করা হয়, তাকে সংখ্যা পদ্ধতি বলা হয়। প্রকৃতপক্ষে সংখ্যা পদ্ধতি হলো সংখ্যা প্রকাশের একটি নিয়ম, যাতে নিচের বিষয়সমূহ থাকতে হবে–

- সংখ্যাকে নির্দিষ্ট প্রতীকের সাহায্যে প্রকাশের সুনির্দিষ্ট নিয়মাবলি।
- সংখ্যার যোগ, বিয়োগ, গুণ, ভাগ ইত্যাদি নির্ণয় করার সুনির্দিষ্ট নিয়মাবলি।
- সংখ্যার বিভিন্ন রূপ যেমন– ভগ্নাংশ, ধনাত্মক, ঋণাত্মক ইত্যাদি প্রকাশের সুনির্দিষ্ট/পরিপূর্ণ নিয়মাবলি।

একটি সংখ্যার দুটি অংশ থাকতে পারে, যথা–

১। পূর্ণাংশ (Integer) এবং ২। ভগ্নাংশ (Fraction)।

র‍্যাডিক্স পয়েন্ট দ্বারা পূর্ণাংশ ও ভগ্নাংশ পৃথক থাকে।

যেমন–

$$(N)_b = \underbrace{d_{n-1}\, d_{n-2} \ldots\ldots d_i \ldots\ldots d_1 d_0}_{\text{Integer portion}} \;\underset{\text{Radix point}}{\uparrow}\; \underbrace{d_{-1}\, d_{-2} \ldots\ldots d_{-f} \ldots\ldots d_{-m}}_{\text{Fractional portion}}$$

এখানে,

N = একটি সংখ্যা (a number)

b = সংখ্যা পদ্ধতির বেস বা র‍্যাডিক্স

n = পূর্ণাংশের অঙ্ক সংখ্যা

m = ভগ্নাংশের অঙ্ক সংখ্যা

d_{n-1} = সর্বোচ্চ গুরুত্বের অঙ্ক (Most significant digit)

d_{-m} = সবচেয়ে কম গুরুত্বের অঙ্ক (Least significant digit)

এবং $0 \le (d_i \text{ or } d_{-f}) \le b - 1$

ডিজিট (Digit) ঃ কোনো সংখ্যা পদ্ধতি লিখে প্রকাশ করার জন্য যে-সব মৌলিক চিহ্ন বা সাংকেতিক চিহ্ন ব্যবহার করা হয় তাকে ডিজিট বা অঙ্ক বলা হয়। যেমন– ডেসিমাল সংখ্যা প্রকাশ করার জন্য 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 ব্যবহার করা হয়।

সংখ্যা পদ্ধতির ভিত্তি বা র‍্যাডিক্স ঃ কোনো সংখ্যা পদ্ধতির ভিত্তি বলতে ঐ সংখ্যা পদ্ধতিতে ব্যবহৃত মোট অঙ্ক বা প্রতীকসমূহের সংখ্যাকে বুঝায়।

যেমন– দশমিক সংখ্যা পদ্ধতির ভিত্তি 10, কারণ এ সংখ্যা পদ্ধতিতে 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 এই 10টি মৌলিক চিহ্ন ব্যবহৃত হয়। অনুরূপভাবে বাইনারি সংখ্যা পদ্ধতির ভিত্তি 2, অক্টাল সংখ্যা পদ্ধতির ভিত্তি 8 এবং হেক্সাডেসিমাল সংখ্যা পদ্ধতির ভিত্তি 16, কোনো সংখ্যা পদ্ধতির যে-কোনো সংখ্যাকে প্রথম বন্ধনীর মধ্যে লিখে ডানপাশে ছোট (সাবস্ক্রিপ্ট) করে সেই সংখ্যা পদ্ধতির ভিত্তি লিখতে হয়। নিচে বিভিন্ন সংখ্যা পদ্ধতির নাম ও বেজসহ উদাহরণ দেওয়া হলো–

সংখ্যা পদ্ধতি	উদাহরণ
দশমিক	$(565)_{10}$
বাইনারি	$(110)_2$
অক্টাল	$(563)_8$
হেক্সাডেসিমাল	$(5C3)_{16}$

বিভিন্ন প্রকার সংখ্যা পদ্ধতি ঃ সাধারণভাবে ব্যবহৃত সংখ্যা পদ্ধতিগুলো হলো–

১। দশমিক সংখ্যা পদ্ধতি (Decimal number system)

২। বাইনারি সংখ্যা পদ্ধতি (Binary number system)

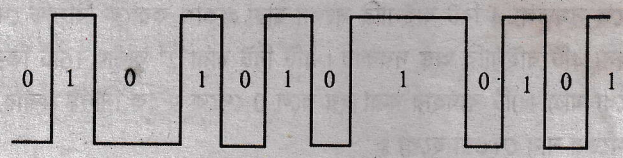
৩। অক্টাল সংখ্যা পদ্ধতি (Octal number system)

৪। হেক্সাডেসিমাল সংখ্যা পদ্ধতি (Hexadecimal number system)

১। দশমিক সংখ্যা পদ্ধতি (Decimal number system) ঃ ডেসিমাল অর্থ দশ। যে পদ্ধতিতে 0 থেকে 9 পর্যন্ত এই দশটি ডিজিট এর মাধ্যমে সংখ্যা প্রকাশ করা হয়, তাকে ডেসিমাল সংখ্যা পদ্ধতি বলে। দশটি প্রতীক 0, 1, 2, 3, 4, 5, 6, 7, 8 এবং 9 ব্যবহার করে ডেসিমাল সংখ্যা পদ্ধতি গড়ে উঠেছে। দশটি প্রতীক ব্যবহৃত হয় বলে এ পদ্ধতিকে দশভিত্তিক পদ্ধতিও বলা হয়। সাধারণ হিসাবনিকাশের জন্য দশমিক সংখ্যা পদ্ধতি ব্যবহার করা হয়। দশমিক সংখ্যা পদ্ধতির ভিত্তি 10 বিধায় এ পদ্ধতিতে অঙ্কগুলোর হ্রাস-বৃদ্ধি ঘটে দশ গুণ করে।

উদাহরণ ঃ $(425)_{10}$

২। বাইনারি সংখ্যা পদ্ধতি (Binary number system) ঃ বাইনারি অর্থ দুই। যে পদ্ধতিতে 0, 1 এই দুইটি ডিজিট এর মাধ্যমে সংখ্যা প্রকাশ করা হয়, তাকে বাইনারি সংখ্যা পদ্ধতি বলে। এ পদ্ধতির 0 অথবা 1 অঙ্ককে সংক্ষেপে বিট (Bit বা Binary digit) বলে। বাইনারি সংখ্যা পদ্ধতি হলো সরলতম গণনা পদ্ধতি। এ পদ্ধতির বিট দুটিকে সহজে ইলেকট্রনিক উপায়ে নির্দিষ্ট করা সম্ভব। তাই ইলেকট্রনিক যন্ত্রে এই পদ্ধতির ব্যবহার সুবিধানজক। ২.১ নং চিত্রে 0 ও 1 এই বিট দুটিকে দুই অবস্থাবিশিষ্ট বিদ্যুৎ তরঙ্গ দ্বারা দেখানো হয়েছে। এখানে 0 ভোল্ট দ্বারা 0 বিট এবং 5 ভোল্ট দ্বারা 1 বিট নির্দেশ করা হয়েছে।



চিত্র ঃ ২.১ দুই অবস্থাবিশিষ্ট সংকেত

বাইনারি সংখ্যা পদ্ধতির ভিত্তি হচ্ছে 2।

উদাহরণ ঃ $(1001.101)_2$

৩। অক্টাল সংখ্যা পদ্ধতি (Octal number system) ঃ অক্টাল অর্থ আট। যে সংখ্যা পদ্ধতিতে 0, 1, 2, 3, 4, 5, 6 এবং 7 এই আটটি অঙ্ক বা চিহ্ন ব্যবহার করে সংখ্যা প্রকাশ করা হয়, তাকে অক্টাল সংখ্যা পদ্ধতি বলা হয়।

এই সংখ্যা পদ্ধতির ভিত্তি আট। আধুনিক কম্পিউটার উন্নয়নের প্রাথমিক অবস্থায় এবং মাইক্রোকম্পিউটারে এ গণনা পদ্ধতি ব্যবহার করা হয়।

উদাহরণ ঃ $(560)_8$

৪। হেক্সাডেসিমাল সংখ্যা পদ্ধতি (Hexadecimal number system) ঃ হেক্সাডেসিমাল অর্থ ষোড়শ বা ষোল। যে সংখ্যা পদ্ধতিতে 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E এবং F এই 16টি অঙ্ক বা চিহ্ন ব্যবহার করে সংখ্যা প্রকাশ করা হয়, তাকে হেক্সাডেসিমাল সংখ্যা পদ্ধতি বলা হয়। এই সংখ্যা পদ্ধতির ভিত্তি 16। ছোট-বড় অনেক কম্পিউটারে এই সংখ্যা পদ্ধতি ব্যবহার করা হয়।

উদাহরণ ঃ $(1A5)_{16}$ একটি হেক্সাডেসিমাল সংখ্যা।

একটি ছকের মাধ্যমে ডেসিমাল, বাইনারি, অক্টাল ও হেক্সাডেসিমাল সংখ্যা পদ্ধতির ভিত্তি/বেজ, ব্যবহৃত ডিজিট এবং উদাহরণ দেয়া হলো–

সংখ্যা পদ্ধতি	ভিত্তি	ব্যবহৃত ডিজিটসমূহ	উদাহরণ
ডেসিমাল সংখ্যা পদ্ধতি	10	0, 1, 2, 3, 4, 5, 6, 7, 8 এবং 9	$(576)_{16}$
বাইনারি সংখ্যা পদ্ধতি	2	0 এবং 1	$(1101)_2$
অক্টাল সংখ্যা পদ্ধতি	8	0, 1, 2, 3, 4, 5, 6 এবং 7	$(732)_8$
হেক্সাডেসিমাল সংখ্যা পদ্ধতি	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9 A, B, C, D, E এবং F	$(35D)_{16}$

২.২ বিসিডি কোড, এক্সেস-3 কোড, গ্রে কোড, হ্যামিং কোড, ইউনিকোড এবং এএসসিআইআই কোডের সং (Define BCD code, Excess-3 code, Gray code, Hamming code, Unicode and ASCII code) ঃ

কোড (Code) ঃ ডিজিটাল কৌশল ব্যবহার করে তথ্য প্রেরণ ও প্রক্রিয়াকরণের জন্য প্রতিটি বর্ণ, সংখ্যা বা বিশেষ চিহ্ন আলাদাভাবে বাইনারি বিট অর্থাৎ 0 বা 1-এ রূপান্তর করে বিভিন্নভাবে সাজিয়ে অদ্বিতীয় সংকেত তৈরি করা হয়। এই অদ্বিতী সংকেতকে কোড বলে। ডেটা ইনপুটের জন্য কোডিং প্রয়োজন। প্রসেসিং শেষে আবার আউটপুটকে ডিকোডিং করা হয়। এ পদ্ধতি কোডকে আবার বর্ণ, সংখ্যা বা চিহ্নে রূপান্তর করা হয়। প্রয়োগের ক্ষেত্র অনুসারে নানা ধরনের কোড আবিস্কৃত হয়েছে। যেমন–

১। অক্টাল কোড (Octal code)
২। হেক্সাডেসিমাল কোড (Hexadecimal code)
৩। বিসিডি কোড (BCD code)
৪। আলফানিউমেরিক কোড (Alphanumeric code)
৫। অ্যাসকি কোড (ASCII code)
৬। ইবিসিডিআইসি কোড (EBCDIC code)
৭। ইউনিকোড (Unicode)
৮। গ্রে-কোড (Gray code)

বর্তমানে অ্যাসকি কোড এবং ইউনিকোড বেশি ব্যবহৃত হয়।

বিসিডি কোড (BCD Code) ঃ BCD শব্দ সংক্ষেপটির পূর্ণ অর্থ হলো Binary Coded Decimal.

দশমিক সংখ্যার প্রতিটি অঙ্ককে সমতুল্য 4 বিট বাইনারি সংখ্যা দ্বারা প্রকাশ করাকে বিসিডি কোড বলে। 0 থেকে 9 এই দশটি অঙ্কের প্রতিটিকে নির্দেশ করার জন্য 4টি বাইনারি অঙ্ক দরকার। 4টি বিট দ্বারা 2^4 অর্থাৎ 16টি ভিন্ন অবস্থা (Combination) নির্দিষ্ট করা যায়। কিন্তু 16টি অবস্থার মধ্যে মাত্র 10টি ব্যবহার করা হয় বলে 0 থেকে 9 কে নির্দিষ্ট করার জন্য কয়েক প্রকার BCD কোড সম্ভব। নিচে পাঁচ প্রকার BCD কোডের নাম দেওয়া হলো ঃ

১। BCD 8421 কোড
২। BCD 5421 কোড
৩। BCD 2421 কোড
৪। BCD 7421 কোড
৫। BCD 642 – 3 কোড।

BCD 8421 কোডকে সাধারণভাবে BCD কোড বলা হয়। এই কোডে বাইনারি 4টি অঙ্কের ডান হতে বামের দিকে প্রথমটির স্থানীয় মান $2^0 = 1$ দ্বিতীয়টির স্থানীয় মান $2^1 = 2$, তৃতীয়টির $2^2 = 4$ এবং চতুর্থটির $2^3 = 8$ অর্থাৎ বাম থেকে ডানে স্থানীয় মান 8, 4, 2, 1; এজন্য এই কোডকে BCD 8421 কোড বলা হয়। BCD 8421 কোড যা BCD কোড বা NBCD (Natural Binary Coded Decimal) কোড নামেও পরিচিত।

নিচের টেবিলে 0 থেকে 9 এই দশটি অঙ্কের BCD দেখানো হলো–

দশমিক অঙ্ক	বিসিডি কোড (8–4–2–1)
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

উদাহরণ-১। $(925)_{10}$ কে বিসিডি কোডে রূপান্তর কর।

সমাধান ঃ

925

1001 0010 0101

$\therefore (925)_{10} = (100100100101)_{BCD}$

উদাহরণ-২। $(25)_{10}$ এর BCD কত?

সমাধান ঃ

2 5

0010 0101

∴ নির্ণেয় BCD 0010 0101

উদাহরণ-৩। 127, 96 এই সংখ্যাগুলোকে (ক) BCD 8421, (খ) BCD 2421, (গ) BCD 642–3 কোডে লিখতে হবে।

সমাধান ঃ

(ক) 127	:	0001	0010	0111
96	:		1001	0110
(খ) 127	:	0001	0010	1101
96	:		1111	1100
(গ) 127	:	0101	0010	1101
96	:		1111	0110

উদাহরণ-৪। $(531)_{10}$ এর BCD কোড নির্ণয় কর।

সমাধান ঃ

5 = 0101

3 = 0011

1 = 0001

∴ $(531)_{10}$ এর BCD কোড = 010100110001

এক্সেস-3 কোড (Excess-3 code) ঃ ডেসিমাল সংখ্যার প্রতিটি অঙ্কের বিসিডি কোডের সাথে 3 যোগ করলে এক্সেস-3 কোড পাওয়া যায়।

যেমন– $(4)_{10}$ এর এক্সেস-3 কোড হলো ঃ

```
  0100  BCD
+ 0011  Add
---------
  0111  Excess-3 code
```

নিচের টেবিলে 0 থেকে 9 এই দশটি অঙ্কের এক্সেস-3 কোড দেখানো হলো–

ডেসিমাল অঙ্ক	BCD Code	Excess-3 Code
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

উদাহরণ-৫। $(54)_{10}$ এর এক্সেস-3 কোড নির্ণয় কর।

সমাধান ঃ

5 এর BCD = 0101
+ 0011
1000

4 এর BCD = 0100
+ 0011
0111

∴ $(54)_{10}$ এর এক্সেস-3 কোড = 10000111 [বাকাশিবো-২০১

উদাহরণ-৬। $(27)_{10}$ এর এক্সেস-3 কোড কত?

সমাধান ঃ

2 এর BCD = 0010
+ 0011
0101

7 এর BCD = 0111
+ 0011
1010

∴ $(27)_{10}$ এর এক্সেস-3 কোড = 01011010

গ্রে-কোড (Gray code) ঃ এটি এমন এক ধরনের কোড, যার মাধ্যমে ডেসিমাল সংখ্যাকে বাইনারি আকারে (For এমনভাবে প্রকাশ করা যায় যেন প্রতিটি গ্রে-কোড সংখ্যা পূর্ববর্তী ও পরবর্তী সংখ্যা হতে মাত্র এক বিট তফাৎ হয়। এটি ওয়েট কোড নয়। যেমন– ডেসিমাল সংখ্যা 5 এর গ্রে কোড 0111 এবং 6 এর গ্রেড কোড 0101; এখানে কোড দুটির মধ্যে মাত্র এক বিট বামদিক থেকে ৩য় বিট তফাৎ।

গ্রে-কোড নিম্নলিখিত উপায়ে গঠন করা যেতে পারে–

১। 1-বিট গ্রে কোডে দুটি কোড শব্দ 0 এবং 1 দ্বারা যথাক্রমে ডেসিমাল সংখ্যা 0 এবং 1 প্রকাশ করা যায়।

২। n-বিটের ($n \geq 2$) গ্রে-কোডের ক্ষেত্রে প্রথম 2^{n-1} কোড শব্দ, $(n-1)$ বিটের গ্রে কোড অগ্রভাগে 0 যুক্ত করে ক্রমানুসা লিখতে হবে।

৩। শেষ 2^{n-1} কোড শব্দ, $(n-1)$ বিটের গ্রে-কোডগুলো অগ্রভাগে 1 যুক্ত করে বিপরীত ক্রমানুসারে (প্রথম 2^{n-1} ও শেষ 2^n কোডের মাঝখানে আয়না স্থাপন করা হলে যেরূপ দেখা যাবে) লিখতে হবে।

উদাহরণ-৭। (ক) 1-বিট (খ) 2-বিট (গ) 3-বিট গ্রে কোড নির্ণয় কর এবং তাদের সমতুল্য ডেসিমাল সংখ্যা দেখাও।

সমাধান ঃ (ক) (১) নং নিয়ম অনুযায়ী 1-বিট গ্রে-কোড ও এর সমতুল্য ডেসিমাল সংখ্যা নিচে দেখানো হলো ঃ

ডেসিমাল সংখ্যা	গ্রে কোড
0	0
1	1

(খ) (২) এবং (৩) নং নিয়ম এবং 1-বিটের গ্রে কোড ব্যবহার করে 2-বিট গ্রে-কোড ও এর সমতুল্য ডেসিমাল সংখ্যা দেখানো হলো ঃ

ডেসিমাল সংখ্যা	গ্রে-কোড
0	0 0
1	0 1
2	1 1
3	1 0

(গ) 2-বিট গ্রে-কোড ব্যবহার করে 3-বিট গ্রে-কোড ও এর সমতুল্য ডেসিমাল সংখ্যা দেখানো হলো ঃ

ডেসিমাল সংখ্যা	গ্রে-কোড
0	0 00
1	0 01
2	0 11
3	0 10
4	1 10
5	1 11
6	1 01
7	1 00

হ্যামিং কোড (Hamming code) ঃ এরর নির্ণয় ও সংশোধনের জন্য ব্যবহৃত কোডের মধ্যে বহুল ব্যবহৃত কোড হলো হ্যামিং কোড। এতে n সংখ্যক ডাটার মধ্যে k সংখ্যক প্যারিটি বিট যুক্ত করে n + k সংখ্যক বিটের ডাটা তৈরি করা হয়। হ্যামিং কোড পজিশন অবস্থানগুলো 1 থেকে n + k এর মধ্যে হয়। এর মধ্যে 2 এর ঘাত আকারে যে মান পাওয়া যায় তা দ্বারা প্যারিটি পজিশন নির্ধারণ করা হয় অর্থাৎ 1, 2, 4, 8, ... 2^{k-1}। আর বাকি পজিশন ডাটার বিটগুলো থাকে। যে-কোনো ডাটাতে এ কোড ব্যবহৃত হয়। হ্যামিং কোডে প্যারিটি বিটের সংখ্যা k এর মান নির্ণয় করার জন্য নিম্নের সম্পর্কে জানা যায়।

$2^k \geq n + k + 1$ ..... (i)

উদাহরণ ঃ ধরি, 7 (সাত) বিটের একটি ডাটা 1011001. এতে প্যারিটি বিট যুক্ত করে হ্যামিং কোড ডাটা নির্ণয় করতে হবে।

সমাধান ঃ

আমরা জানি, হ্যামিং কোডে প্যারিটি বিটের সংখ্যা নির্ণয়ের সম্পর্ক–

$2^k \geq n + k + 1$ ..... (ii)

এখানে, n = 7

k = ?

(ii) নং সমীকরণ থেকে দেখা যায় k এর সর্বনিম্ন মান 4 হলে সম্পর্কটি সমর্থন করে।

অর্থাৎ, $2^4 > (7 + 4 + 1)$

বা, 16 > 12

প্যারিটি বিট সংখ্যা 4টি। মোট হ্যামিং কোড বিট (7 + 4) = 11

প্যারিটি বিটসমূহ যথাক্রমে ১ম, ২য়, ৪র্থ ও ৮ম পজিশনে যুক্ত করে নিম্নরূপ হ্যামিং কোড ডাটা হবে।

10	9	8	7	6	5	4	3	2	1	→ বিট পজিশন
0	1	P_4	1	0	0	P_3	1	P_2	P_1	→ হ্যামিং পজিশন

প্যারিটি বিটসমূহ জোড় প্যারিটি বা বিজোড় প্যারিটি হতে পারে। প্যারিটি বিটগুলো নির্ণয় করার জন্য যে অবস্থানে 1 আছে তা যোগ করা হয়। এক্ষেত্রে 3, 7, 9 এবং 11 নং স্থানে 1 রয়েছে। জোড় প্যারিটির ক্ষেত্রে 1 বিট অবস্থান সংখ্যাগুলোকে বাইনারিতে রূপান্তর করে যোগ করলে প্যারিটি বিটগুলো পাওয়া যাবে। তবে যোগ করার সময় হাতের অঙ্ককে বাদ দিতে হবে অর্থাৎ 1 এর সংখ্যা জোড় হলে 0 হবে।

এখানে,

3 = 0 0 1 1
7 = 0 1 1 1
9 = 1 0 0 1
11 = 1 0 1 1

0 1 1 0
↓ ↓ ↓ ↓
P_4 P_3 P_2 P_1

অর্থাৎ, $P_1 = 0, P_2 = 1, P_3 = 1$, এবং $P_4 = 0$

এ বিটগুলো P এর স্থলে বসিয়ে হ্যামিং কোড পাওয়া যাবে, সুতরাং নির্ণেয় হ্যামিং কোড–

10	9	8	7	6	5	4	3	2	1	→ বিট পজিশন
0	1	0	1	0	0	1	1	1	0	→ হ্যামিং পজিশন

রিসিভারে এ কোডের যে-সব স্থানে 1 আছে সে-সব স্থানের সংখ্যা মানকে বাইনারিতে রূপান্তর করে তা আবার যোগ করে চেক করে দেখতে হবে। তবে যোগ করার সময় হাতের অঙ্কটি বাদ দিতে হবে। যোগফল 0 কি না। যদি যোগফল 0 হয় তবে বুঝা যায় যে, গৃহীত ডাটাতে কোনো ভুল নাই নতুবা ভুল আছে।

ইউনিকোড (Unicode) ঃ ইউনিকোডের পূর্ণ অর্থ Universal Code বা সার্বজনীন কোড। ইউনিকোড হচ্ছে 16 বিটের কোড। এ কোডের মাধ্যমে 2^{16} = 65536টি অদ্বিতীয় চিহ্নকে নির্দিষ্ট করা যায়। বিশ্বের ছোট বড় সকল ভাষাকে কম্পিউটারে কোডভুক্ত করার জন্য ইউনিকোড ব্যবহৃত হয়। বর্তমানে বিশ্বব্যাপী প্রচলিত আসকি কোডের পাশাপাশি ইউনিকোড সিস্টেম চালু হয়েছে। বিশ্বের সব ভাষাভাষীর জন্য তাদের ভাষায় কম্পিউটিং করা সহজ করার লক্ষ্যে ১৯৯১ সালে অ্যাপেল কম্পিউটার কর্পোরেশন এবং Xerox Corporation এর একদল কম্পিউটার প্রকৌশলী যৌথভাবে ইউনিকোড উদ্ভাবন করেন। ইউনিকোড ASCII কোডের সাথে কম্প্যাটিবল। অর্থাৎ ইউনিকোডের প্রথম 256টি কোড অ্যাসকি 256টি কোডের অনুরূপ।

অ্যাসকি কোড (ASCII CODE) ঃ অ্যাসকি কোডের পূর্ণ অর্থ হচ্ছে American Standard Code for Infor Interchange। এটি একটি Alphanumeric কোড। এটি 7 বিটের কোড। এই কোডের 7 বিটের মাধ্যমে $2^7 = 128$টি চিহ্নকে করা যায়। অ্যাসকি কোডের ফরম্যাট হলো–

$X_6 X_5 X_4 X_3 X_2 X_1 X_0$; যেখানে X এর মান 0 অথবা 1 হতে পারে। বামদিকে 3টি বিটকে জোন এবং ডানদিকের 4টি সংখ্যা সূচক বিট হিসেবে ধরা হয়। কম্পিউটার এবং ইনপুট, আউটপুট এর জন্য ব্যবহৃত যন্ত্র যেমন– কী-বোর্ড, মাউস, প্রিন্টার ইত্যাদির মধ্যে আলফানিউমেরিক তথ্য আদান-প্রদানে অ্যাসকি কোড ব্যবহৃত হয়। ১৯৬৫ সালে রবার্ট বিমার 7 অ্যাসকি কোড উদ্ভাবন করেন।

২.৩ এক কোড হতে অন্য কোডে রূপান্তর অনুশীলনী (Practice the conversion of one code to another

এক সংখ্যা পদ্ধতি থেকে অন্য সংখ্যা পদ্ধতিতে রূপান্তর করাকে বলা হয় সংখ্যা পদ্ধতির রূপান্তর। আমরা দৈনন্দিন দশমিক সংখ্যা পদ্ধতি ব্যবহার করি কিন্তু কম্পিউটারে বাইনারি, অক্টাল বা হেক্সাডেসিমাল সংখ্যা পদ্ধতি ব্যবহৃত হয়। কম্পিউটারের অভ্যন্তরীণ প্রক্রিয়াকরণ বুঝার জন্য বিভিন্ন সংখ্যা পদ্ধতির রূপান্তরগুলো জানা প্রয়োজন।

দশমিক সংখ্যা থেকে যে-কোনো সংখ্যা পদ্ধতিতে রূপান্তর ঃ

দশমিক সংখ্যা (Decimal number)

- **বাইনারি (Binary) ঃ** বাইনারি সংখ্যার বেজ 2 দ্বারা দশমিক সংখ্যার পূর্ণ অংশকে ভাগ এবং ভগ্নাংশে গুণ করতে হবে।
- **অক্টাল (Octal) ঃ** অক্টাল সংখ্যার বেজ 8 দ্বারা দশমিক সংখ্যার পূর্ণ অংশকে ভাগ এবং ভগ্নাংশকে গুণ করতে হবে।
- **হেক্সাডেসিমাল (Hexadecimal) ঃ** হেক্সাডেসিমাল সংখ্যার বেজ 16 দ্বারা দশমিক সংখ্যার পূর্ণ অংশকে ভাগ এবং ভগ্নাংশকে গুণ করতে হবে।

দশমিক সংখ্যাকে বাইনারি সংখ্যায় রূপান্তর ঃ

পূর্ণাংশের ক্ষেত্রে–

দশমিক পূর্ণসংখ্যাকে বাইনারিতে রূপান্তরের জন্য ভাগফল 0 না হওয়া পর্যন্ত সংখ্যাটিকে অনবরত 2 দ্বারা ভাগ করার ভাগশেষগুলোকে সাজিয়ে সংখ্যাটির সমকক্ষ বাইনারি মান পাওয়া যায়। ভাগশেষগুলোকে সাজানোর সময় সর্বশেষ ভাগশেষ শুরু করে ক্রমান্বয়ে প্রথম ভাগশেষ পর্যন্ত লিখতে হবে। এক্ষেত্রে সর্বশেষ ভাগশেষকে সর্বোচ্চ গুরুত্বের বিট (MSB) এবং প্রথম ভাগশেষকে সর্বনিম্ন গুরুত্বের বিট (LSB) পাওয়া যাবে।

উদাহরণ-১। নিম্নের দশমিক সংখ্যাগুলোকে বাইনারি সংখ্যায় রূপান্তর কর।

(ক) $(14)_{10}$ (খ) $(21)_{10}$

সমাধান ঃ

(ক) ভাজক ভাজ্য/ভাগফল ভাগশেষ

```
2 | 14
2 | 7 –  0     LSB
  2 | 3 –  1
    2 | 1 –  1
        0 –  1   MSB
```

$\therefore (14)_{10} = (1110)_2$

	ভাগফল	ভাগশেষ
2	21	
2	10 –	1
2	5 –	0
2	2 –	1
2	1 –	0
	0 –	1

LSB ↑ MSB

$\therefore (21)_{10} = (10101)_2$

ভগ্নাংশের ক্ষেত্রে–

দশমিক ভগ্নাংশকে বাইনারিতে রূপান্তরের জন্য ভগ্নাংশকে 2 দ্বারা অনবরত গুণ করতে হবে যতক্ষণ পর্যন্ত গুণফলে দশমিক বিন্দুর ডানপার্শ্বে 0 বা পূর্বের কোনো সংখ্যা পুনরাবৃত্তি না হয়। প্রতিক্ষেত্রে গুণফলের পূর্ণ অংশ আলাদা করতে হবে। যদি পূর্ণ অংশ হয় 0 পৃথক করতে হবে। তবে যদি চার-পাঁচ বার গুণ করার পর গুণফলে দশমিক বিন্দুর ডানপার্শ্বে শূন্য বা পূর্বের কোনো সংখ্যা পুনরাবৃত্তি না হয় তাহলে আসন্ন মান হিসেবে রাখা যেতে পারে। গুণফলের পূর্ণ অঙ্কগুলোকে প্রথম থেকে শেষ পর্যন্ত ক্রমান্বয়ে লিখলে ভগ্নাংশের সমকক্ষ বাইনারি সংখ্যা পাওয়া যাবে।

উদাহরণ-২। $(0.375)_{10}$ কে বাইনারিতে রূপান্তর কর।

সমাধান ঃ

পূর্ণাংশ	ভগ্নাংশ
	.375
	× 2
0	.750
	× 2
1	.500
	× 2
1	.0

$\therefore (0.375)_{10} = (0.011)_2$

উদাহরণ-৩। $(38.85)_{10}$ সংখ্যাটিকে বাইনারি সংখ্যায় রূপান্তর কর।

সমাধান ঃ পূর্ণ অংশ $(38)_{10}$ কে বাইনারিতে রূপান্তর ঃ

	ভাগফল	ভাগশেষ
2	38	
2	19 –	0
2	9 –	1
2	4 –	1
2	2 –	0
2	1 –	0
	0 –	1

LSB ↑ MSB

$\therefore (38)_{10} = (100110)_2$

ভগ্নাংশ $(0.85)_{10}$ কে বাইনারিতে রূপান্তর ঃ

সমাধান ঃ

পূর্ণাংশ	ভগ্নাংশ
	.85
	× 2
1	.7
	× 2
1	.4
	× 2
0	.8
	× 2
1	.6
	× 2
1	.2
	× 2
0	.4

$\therefore (0.85)_{10} = (0.110110 \ldots\ldots)_2$

$\therefore (38.85)_{10} = (100110.110110\ldots\ldots)_2$

দশমিক সংখ্যাকে অক্টাল সংখ্যায় রূপান্তর ঃ দশমিক পূর্ণ সংখ্যাকে 8 দ্বারা অনবরত ভাগ করতে হবে যতক্ষণ পর্যন্ত ভাগফল না হয়। প্রতিক্ষেত্রে ভাগশেষগুলোকে আলাদা করে সংরক্ষণ করতে হবে। পরিশেষে ভাগশেষসমূহকে শেষ থেকে শুরু ক্রমান্বয়ে সাজিয়ে লিখলে অক্টাল সংখ্যা পাওয়া যাবে। দশমিক ভগ্নাংশকে 8 দ্বারা অনবরত গুণ করতে হবে যতক্ষণ পর্যন্ত গুণফল ভগ্নাংশে 0 বা পূর্বের কোনো সংখ্যা পুনাবৃত্তি না হয়। প্রতিক্ষেত্রে গুণফলের পূর্ণ অংশকে আলাদা করতে হবে। গুণফলের অংশসমূহকে শুরু থেকে শেষ পর্যন্ত ক্রমান্বয়ে সাজিয়ে লিখলে ভগ্নাংশে অক্টাল সংখ্যা পাওয়া যাবে।

উদাহরণ-8 । (ক) $(269)_{10}$ (খ) $(0.46)_{10}$ সংখ্যা দুটিকে অক্টাল সংখ্যায় রূপান্তর কর।

সমাধান ঃ

(ক)

```
8 | 269
  8 | 33 –   5   ↑ LSB
    8 | 4 –   1
        0 –   4   MSB
```

$\therefore (269)_{10} = (415)_8$

(খ)

পূর্ণাংশ	ভগ্নাংশ
	.46
	× 8
3	.68
	× 8
5	.44
	× 8
3	.52
	× 8
4	.16

$\therefore (.46)_{10} = (.3534\ldots)_8$

উদাহরণ-৫। $(175.15)_{10}$ সংখ্যাকে অক্টাল সংখ্যায় রূপান্তর কর।

সমাধান ঃ পূর্ণ সংখ্যার ক্ষেত্রে ঃ

```
8 | 175
  8 | 21 –   7   ↑ LSB
    8 | 2 –   5   |
        0 –   2   | MSB
```

$\therefore (175)_{10} = (257)_8$

ভগ্নাংশের ক্ষেত্রে ঃ

পূর্ণাংশ	ভগ্নাংশ
	.15
	× 8
1	.20
	× 8
1	.60
	× 8
4	.80
	× 8
6	.40
	× 8
3	.20

$\therefore (.15)_{10} = (.11463 .....)_8$

$\therefore (175.15)_{10} = (257.11463 .....)_8$

● **দশমিক সংখ্যাকে হেক্সাডেসিমাল সংখ্যায় রূপান্তর ঃ** দশমিক পূর্ণ সংখ্যাকে 16 দ্বারা অনবরত ভাগ করতে হবে। যতক্ষণ পর্যন্ত ভাগফল 0 না হয়। প্রতিক্ষেত্রে ভাগশেষগুলোকে আলাদা করে সংরক্ষণ করতে হবে। পরিশেষে ভাগশেষসমূহকে শেষ থেকে শুরু পর্যন্ত ক্রমান্বয়ে সাজিয়ে লিখলে হেক্সাডেসিমাল সংখ্যা পাওয়া যাবে। দশমিক ভগ্নাংশকে 16 দ্বারা অনবরত গুণ করতে হবে যতক্ষণ পর্যন্ত গুণফলে ভগ্নাংশ 0 বা পূর্বের কোনো ভগ্নাংশ পুনরাবৃত্তি না হয়। প্রতিক্ষেত্রে গুণফলের পূর্ণ অংশ আলাদা করতে হবে। গুণফলের পূর্ণ অংশসমূহকে শুরু থেকে শেষ পর্যন্ত ক্রমান্বয়ে সাজিয়ে শিখলে ভগ্নাংশে হেক্সাডেসিমাল সংখ্যা পাওয়া যাবে।

উদাহরণ-৬। (ক) $(269)_{10}$ (খ) $(.625)_{10}$ (গ) $(2479.50)_{10}$ সংখ্যাগুলোকে হেক্সাডেসিমাল সংখ্যায় রূপান্তর কর।

সমাধান ঃ

(ক)
```
16 | 269
   16 | 16 –   13 (D) ↑ LSB
      16 | 1 –   0     |
           0 –   1     | MSB
```

$\therefore (269)_{10} = (10D)_{16}$

(খ)

পূর্ণাংশ	ভগ্নাংশ
	.625
	× 16
A = 10	.00

$\therefore (.625)_{10} = (.A)_{16}$

(গ) পূর্ণ সংখ্যার ক্ষেত্রে ঃ

```
16 | 2479
   16 | 154 –   15 = F ↑
      16 | 9 –   10 = A |
           0 –   9     |
```

$\therefore (2479)_{10} = (9AF)_{16}$

$\therefore (2479.50)_{10} = (9AF.8)_{16}$

ভগ্নাংশের ক্ষেত্রে ঃ

পূর্ণসংখ্যা	ভগ্নাংশ
	.50
	× 16
↓8	.00

$\therefore (.50)_{10} = (.8)_{16}$

যে-কোনো সংখ্যা পদ্ধতি থেকে দশমিক সংখ্যা পদ্ধতিতে রূপান্তরের সাধারণ নিয়ম–

বাইনারি, অক্টাল বা হেক্সাডেসিমাল সংখ্যা পদ্ধতি থেকে দশমিক সংখ্যা পদ্ধতিতে রূপান্তরের ক্ষেত্রে নিম্নলিখিত ধাপ অনুসরণ করতে হবে–

ধাপ-১ ঃ প্রদত্ত সংখ্যাটির বেজ শনাক্ত করে সংখ্যাটির অন্তর্গত প্রত্যেকটি অঙ্কের স্থানীয় মান বের করতে হবে।

ধাপ-২ ঃ সংখ্যার অন্তর্গত প্রত্যেকটি অঙ্কের নিজস্ব মানকে তার স্থানীয় মান দিয়ে গুণ করতে হবে।

ধাপ-৩ ঃ গুণফলের যোগফলই হবে সমতুল্য দশমিক সংখ্যা।

দশমিক সংখ্যায় রূপান্তরের সাধারণ সম্পর্ক হলো–

$$(N)_b = (d_{n-1} \times b^{n-1} + d_{n-2} \times b^{n-2} + d_{n-3} \times b^{n-3} + \text{--------} + d_2 \times b^2 + d_1 \times b^1 + d_0 \times b^0 + d_{-1} \times b^{-1} + d_{-2} \times b^{-2} + d_{-3} \times b^{-3} + \text{----} + d_{-m} \times b^{-m}$$

এখানে N = একটি সংখ্যা

b = সংখ্যা পদ্ধতিটির বেস

n = পূর্ণ অংশের অঙ্ক সংখ্যা

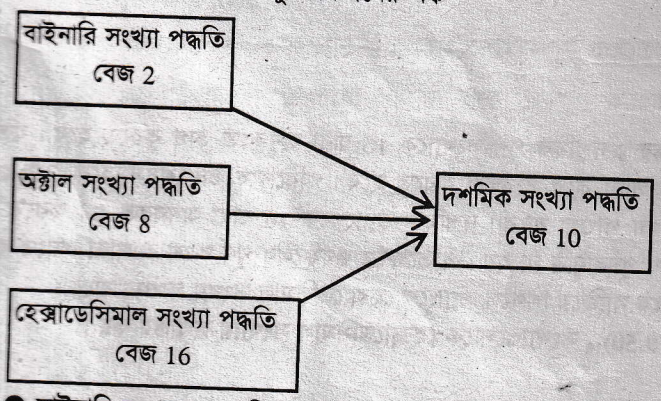
m = ভগ্নাংশের অঙ্ক সংখ্যা

d_{n-1} = সর্বোচ্চ গুরুত্বের অঙ্ক (MSB)

d_0 = পূর্ণ অংশের সর্বনিম্ন গুরুত্বের অঙ্ক

d_{-1} = দশমিক বিন্দুর ডানের প্রথম অঙ্ক

d_{-m} = দশমিক বিন্দুর সর্বডানের অঙ্ক



● **বাইনারি সংখ্যাকে দশমিক সংখ্যায় রূপান্তর ঃ** বাইনারি সংখ্যার প্রতিটি অঙ্ককে এর স্থানীয় মান দিয়ে গুণ করে যোগ করলে সমতুল্য ডেসিমাল সংখ্যা পাওয়া যাবে। পূর্ণ অংশের ক্ষেত্রে দশমিক বিন্দুর আগে ডানদিক থেকে বামদিকে প্রথম অঙ্কের স্থানীয় মান 2^0, দ্বিতীয় অঙ্কের স্থানীয় মান 2^1, তৃতীয় অঙ্কের স্থানীয় মান 2^2, চতুর্থ অঙ্কের স্থানীয় মান 2^3 এভাবে নির্ণয় করা যায়।

আবার ভগ্নাংশের ক্ষেত্রে দশমিক বিন্দুর পর বামদিক থেকে ডানদিকে প্রথম অঙ্কের স্থানীয় মান 2^{-1}, দ্বিতীয় অঙ্কের স্থানীয় মান 2^{-2}, তৃতীয় অঙ্কের স্থানীয় মান 2^{-3}, চতুর্থ অঙ্কের স্থানীয় মান 2^{-4} এভাবে নির্ণয় করা যায়। বাইনারি সংখ্যা পদ্ধতির ভিত্তি 2। তাই পূর্ণ অংশের ক্ষেত্রে 2 এর ধনাত্মক ঘাত এবং ভগ্নাংশের ক্ষেত্রে 2 এর ঋণাত্মক ঘাত দিয়ে হিসাব করতে হয়।

উদাহরণ-৭। নিম্নের বাইনারি সংখ্যাগুলোর দশমিক মান নির্ণয় কর ঃ

(ক) $(10110)_2$ (খ) $(0.1011)_2$ (গ) $(101.111)_2$

সমাধান ঃ (ক) $(10110)_2 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

$= 16 + 0 + 4 + 2 + 0$

$= (22)_{10}$ **(উত্তর)**

(খ) $(0.1011)_2 = 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4}$

$= \frac{1}{2} + 0 + \frac{1}{8} + \frac{1}{16}$

$= 0.5 + 0 + 0.125 + 0.0625$

$= (0.6875)_{10}$ **(উত্তর)**

(গ) $(101.111)_2 = 1\times2^2 + 0\times2^1 + 1\times2^0 + 1\times2^{-1} + 1\times2^{-2} + 2\times2^{-3}$

$= 4 + 0 + 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8}$

$= 5 + 0.5 + 0.25 + 0.125$

$= (5.875)_{10}$ **(উত্তর)**

উদাহরণ-৮। $(101101.10101)_2$ বাইনারি সংখ্যাকে ডেসিমাল সংখ্যায় রূপান্তর কর।

সমাধান ঃ $(101101.10101)_2 = 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times\times2^2 + 0\times2^1 + 1\times2^0 + 1\times2^{-1} +$
$0\times2^{-2} + 1\times2^{-3} + 0\times2^{-4} + 1\times2^{-5}$

$= 32 + 0 + 8 + 4 + 0 + 1 + \frac{1}{2} + 0 + \frac{1}{8} + 0 + \frac{1}{32}$

$= (45.65625)_{10}$ **(উত্তর)**

● **অক্টাল সংখ্যাকে দশমিক সংখ্যায় রূপান্তর ঃ** অক্টাল সংখ্যার প্রতিটি অংককে এর স্থানীয় মান দিয়ে গুণ করে যোগ করলে সমকক্ষ ডেসিমাল সংখ্যা পাওয়া যাবে। পূর্ণ অংশের ক্ষেত্রে দশমিক বিন্দুর আগে ডানদিক থেকে বামদিকে প্রথম অঙ্কের স্থানীয় মান ১, ২য় অঙ্কের স্থানীয় মান 8^1, ৩য় অঙ্কের স্থানীয় মান 8^2, চতুর্থ অঙ্কের স্থানীয় মান 8^3 এভাবে নির্ণয় করা যায়। আবার ভগ্নাংশের ক্ষেত্রে দশমিক বিন্দুর পর বামদিক থেকে ডানদিকে ১ম অঙ্কের স্থানীয় মান 8^{-1}, ২য় অঙ্কের স্থানীয় মান 8^{-2}, ৩য় অঙ্কের স্থানীয় মান 8^{-3}, ৪র্থ অঙ্কের স্থানীয় মান 8^{-4} এভাবে নির্ণয় করা যায়। অক্টাল সংখ্যা পদ্ধতির ভিত্তি ৪। তাই পূর্ণ অংশের ক্ষেত্রে ৪ এর ধনাত্মক ঘাত এবং ভগ্নাংশের ক্ষেত্রে ৪ এর ঋণাত্মক ঘাত দিতে হিসেব করতে হয়।

উদাহরণ-৯। $(34)_8$ (খ) $(0.56)_8$ এবং (গ) $(13.52)_8$ সংখ্যাগুলোকে দশমিক সংখ্যায় রূপান্তর কর।

সমাধান ঃ

(ক) $(34)_8 = 3\times8^1 + 4\times8^0 = 24 + 4 = (28)_{10}$

(খ) $(0.56)_8 = 5\times8^{-1} + 6\times8^{-2}$

$= \frac{5}{8} + \frac{6}{64}$

$= \frac{40+6}{64} = \frac{46}{64} = (0.71875)_{10}$

(গ) $(13.52)_8 = 1\times8^1 + 3\times8^0 + 5\times8^{-1} + 2\times8^{-2}$

$= 8 + 3 + \frac{5}{8} + \frac{2}{64}$

$= 8 + 3 + .625 + .03125$

$= (11.65625)_{10}$

হেক্সাডেসিমাল সংখ্যাকে দশমিক সংখ্যায় রূপান্তর ঃ হেক্সাডেসিমাল সংখ্যার প্রতিটি অঙ্ককে এর স্থানীয় মান দিয়ে গুণ করে যোগ করলে সমকক্ষ ডেসিমাল সংখ্যা পাওয়া যাবে। পূর্ণ অংশের ক্ষেত্রে দশমিক বিন্দুর আগে ডানদিক থেকে বামদিকে ১ম অঙ্কের স্থানীয় মান 16^0, ২য় অঙ্কের স্থানীয় মান 16^1, ৩য় অঙ্কের স্থানীয় মান 16^2, ৪র্থ অঙ্কের স্থানীয় মান 16^3 এভাবে নির্ণয় করা হয়। আবার ভগ্নাংশের ক্ষেত্রে দশমিক বিন্দুর পর বামদিক থেকে ডানদিকে ১ম অঙ্কের স্থানীয় মান 16^{-1}, ২য় অঙ্কের স্থানীয় মান 16^{-2}, ৩য় অঙ্কের স্থানীয় মান 16^{-3}, ৪র্থ অঙ্কের স্থানীয় মান 16^{-4} এভাবে নির্ণয় করা হয়। হেক্সাডেসিমাল সংখ্যার ভিত্তি 16। তাই পূর্ণ অংশের ক্ষেত্রে 16 এর ঘাত ধনাত্মক এবং ভগ্নাংশের ক্ষেত্রে 16 এর ঘাত ঋণাত্মক দিয়ে হিসাব করতে হয়।

উদাহরণ-১০। (ক) $(350)_{16}$ (খ) $(1AC)_{16}$ (গ) $(9AF.8)_{16}$ সংখ্যাগুলোকে দশমিক সংখ্যায় রূপান্তর কর।

সমাধান ঃ

(ক) $(350)_{16} = 3\times16^2 + 5\times16^1 + 0\times16^0$

$= 768 + 80 + 0 = (848)_{10}$

(খ) $(1AC)_{16} = 1\times16^2 + A\times16^1 + C\times16^0$

$= 1\times256 + 10\times16 + 12\times1$

$= 256 + 160 + 12$

$= (428)_{10}$

(গ) $(9AF.8)_{16} = 9\times16^2 + A\times16^1 + F\times16^0 + 8\times16^{-1}$

$= 9\times256 + 10\times16 + 15\times1 + \frac{8}{16}$

$= 2304 + 160 + 15 + 0.50$

$= (2479.50)_{10}$

● **অক্টাল সংখ্যাকে বাইনারি সংখ্যায় রূপান্তর ঃ** অক্টাল সংখ্যার প্রতিটি অঙ্ককে 3 বিটের বাইনারি সংখ্যায় রূপান্তর করলে অ সমকক্ষ বাইনারি সংখ্যা পাওয়া যায়।

উদাহরণ-১১। (ক) $(173)_8$ (খ) $(.045)_8$ (গ) $(17.43)_8$ সংখ্যাগুলোকে বাইনারি সংখ্যায় রূপান্তর কর।

সমাধান ঃ

(ক) 173 → 001 111 011

(খ) .045 → .000 100 101

(গ) 17.43 → 001 111 . 100 011

$\therefore (173)_8 = (001111011)_2$ $\therefore (0.045)_8 = (0.000100101)_2$ $\therefore (17.43)_8 = (001111.100011)_2$

● **বাইনারি সংখ্যাকে অক্টাল সংখ্যায় রূপান্তর ঃ** বাইনারি সংখ্যাকে অক্টালে রূপান্তর করতে হলে পূর্ণ সংখ্যার জন্য ডানদি থেকে বামদিকে এবং ভগ্নাংশ সংখ্যার জন্য বামদিক থেকে ডানদিকে প্রতি তিন বিট একত্রে নিয়ে ছোট ছোট ভাগ করতে হ পূর্ণসংখ্যার ক্ষেত্রে বামদিকের ভাগে খালি থাকলে প্রয়োজনীয় 0 দিয়ে পূর্ণ করতে হয় এবং ভগ্নাংশের ক্ষেত্রে ডানদিকের ভাগে খ থাকলে ডানদিকে প্রয়োজনীয় 0 দিয়ে পূর্ণ করতে হয়। এরপর প্রতিটি ভাগকে তার সমকক্ষ অক্টাল অঙ্ক দিয়ে নির্দিষ্ট করতে হবে এভাবে বাইনারি সংখ্যাকে অক্টালে রূপান্তর করা হয়।

উদাহরণ-১২। (ক) $(1111011)_2$ (খ) $(.1111011)_2$ (গ) $(10011.10)_2$ সংখ্যাগুলোকে অক্টাল সংখ্যায় রূপান্তর কর।

সমাধান ঃ (ক) $(1111011)_2 = 001\ 111\ 011 = (173)_8$

(খ) $(.1111011)_2 = .111\ 101\ 100 = (.754)_8$

(গ) $(10011.10)_2 = 010\ 011.100 = (23.4)_8$

● **হেক্সাডেসিমাল সংখ্যাকে বাইনারি সংখ্যায় রূপান্তর ঃ** হেক্সাডেসিমাল সংখ্যার প্রতিটি অঙ্ককে চার বিটের বাইনারি সংখ্য রূপান্তর করলে সংখ্যাটির সমকক্ষ বাইনারি সংখ্যা পাওয়া যায়। নিচে হেক্সাডেসিমাল অঙ্কের বাইনারি মান দেয়া হলো ঃ

হেক্সাডেসিমাল	বাইনারি	হেক্সাডেসিমাল	বাইনারি
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

উদাহরণ-১৩। (ক) $(17C)_{16}$ (খ) $(0.0F5)_{16}$ (গ) $(CD.43)_{16}$ সংখ্যাগুলোকে বাইনারি সংখ্যায় রূপান্তর কর।

সমাধান ঃ

(ক) $(17C)_{16} = 0001\ 0111\ 1100$
$= (000101111100)_2$

(খ) $(0.0F5)_{16} = (0.0000\ 1111\ 0101)$
$= (0.000011110101)_2$

(গ) $(CD.43)_{16} = 1100\ 1101\ .\ 0100\ 0011$
$= (11001101.01000011)_2$

● **বাইনারি সংখ্যাকে হেক্সাডেসিমাল সংখ্যায় রূপান্তর ঃ** বাইনারি সংখ্যাকে হেক্সাডেসিমাল সংখ্যায় রূপান্তর করতে হলে পূ সংখ্যার জন্য ডানদিক থেকে বামদিকে এবং ভগ্নাংশ সংখ্যার জন্য বামদিক থেকে ডানদিকে প্রতি চার বিট একত্রে নিয়ে ছোট ছো ভাগ করতে হয়। পূর্ণ সংখ্যার ক্ষেত্রে বামদিকের ভাগে খালি থাকলে প্রয়োজনীয় শূন্য দিয়ে পূর্ণ করতে হয় এবং ভগ্নাংশের ক্ষে ডানদিকের ভাগে খালি থাকলে ডানদিকে প্রয়োজনীয় 0 দিয়ে পূর্ণ করতে হয়। এরপর প্রতিটি ভাগকে তার সমকক্ষ হেক্সাডেসিমা ডিজিট দিয়ে নির্দিষ্ট করতে হবে। এভাবে বাইনারি সংখ্যাকে হেক্সাডেসিমাল সংখ্যায় রূপান্তর করা হয়।

২.৫ প্যারিটি ও হ্যামিং কোড (Parity checked code and hamming code) ঃ

প্যারিটি বিট (Parity bit) ঃ কোনো তথ্য স্থানান্তরের সময় বিভিন্ন কারণে এর বাইনারি কোড পরিবর্তন হওয়ার আশ প্রেরিত তথ্যের বাইনারি কোডের ত্রুটি শনাক্তকরণের সহজতম উপায়ই হলো প্যারিটি বিট এর ব্যবহার। প্যারিটি বিট হ অতিরিক্ত বিট, যা একটি কোড গ্রুপের সাথে এক স্থান হতে অন্য স্থানে পাঠানো হয়। এ প্যারিটি বিট 1 বা 0 হতে পারে। নির্ভর করে কোড গ্রুপে কতটি 1 আছে তার উপর।

প্যারিটি দুই প্রকার, যথা– (১) জোড় প্যারিটি এবং (২) বিজোড় প্যারিটি।

জোড় প্যারিটি (Even parity) ঃ যখন কোনো কোড গ্রুপের সর্বমোট 1 এর সংখ্যা প্যারিটি বিটসহ জোড় হয় জ জোড় প্যারিটি বলা হয়। ধরা যাক, কোনো তথ্যের কোড গ্রুপে 10000110 আছে। এতে সর্বমোট তিনটি 1 আছে। এক জোড় প্যারিটি করতে হলে এর সাথে একটি 1 সংযোগ করতে হবে। ফলে এটা হবে–

1100000110
প্যারিট বিট

আর যদি কোড গ্রুপ 1100000 হয় এবং এটাকে জোড় প্যারিটি করতে হলে এর বামে 0 সংযোগ করতে হবে। কার জোড় সংখ্যক অর্থাৎ দুটি 1 আছে।

বিজোড় প্যারিটি (Odd parity) ঃ যখন কোনো তথ্যের কোড গ্রুপে সর্বমোট 1 এর সংখ্যা প্যারিটি বিটসহ বিজোড় স তখন তাকে বিজোড় প্যারিটি বলা হয়। ধরা যাক, কোনো তথ্যের কোড গ্রুপ 1011001, এতে সর্বমোট চারটি 1 আছে এটিকে বিজোড় প্যারিটি করতে হলে এর বামে একটি 1 সংযোগ করতে হবে। অনুরূপভাবে, কোনো কোড গ্রুপে এর সংখ্যা হলে এর সাথে 0 যোগ করতে হবে।

কোনো ডাটা এক স্থান হতে অন্য কোন স্থানে অর্থাৎ ট্রান্সমিটার থেকে রিসিভারে ডাটা প্রেরণ করলে রিসিভারে যদি সেই পাওয়া যায় তাহলে বুঝতে হবে ডাটাতে ভুল আছে। এ ধরনের ভুল নির্ণয়ের জন্য প্যারিটি বিট ব্যবহৃত হয়।

হামিং কোড (Hamming code) ঃ এরর নির্ণয় ও সংশোধনের জন্য ব্যবহৃত কোডের মধ্যে বহুল ব্যবহৃত কোড হলে কোড। এতে n সংখ্যক ডাটার মধ্যে k সংখ্যক প্যারিটি বিট যুক্ত করে n + k সংখ্যক বিটের ডাটা তৈরি করা হয়। এ পজিশন/অবস্থানগুলো 1 থেকে n + k এর মধ্যে হয়। এর মধ্যে 2 এর ঘাত আকারে যে মান পাওয়া যায় তা দ্বারা প্যারিটি পজিশন নির্ধারণ করা হয়, অর্থাৎ 1, 2, 4, 8 ইত্যাদি। আর বাকি পজিশন ডাটার বিটগুলো থাকে। যে-কোনো মানের/ ডাটাতে এ কোড ব্যবহৃত হয়।

উদাহরণ ঃ ধরি 7 (সাত) বিটের একটি ডাটা 1011001, এতে প্যারিটি বিট যুক্ত করা হয়েছে, যার পজিশন যথাক্রমে ১ ৪র্থ ও ৮ম। ফলে ডাটাটি হয় নিম্নরূপ ঃ

1	0	1	P_4	1	0	0	P_3	1	P_2	P_1	
11	10	9	8	7	6	5	4	3	2	1	→বিট পজিশন

হামিং বিটগুলো নির্ণয় করার জন্য কোডের যে-সব স্থানে 1 আছে তা শনাক্ত করা হয়। এ ক্ষেত্রে 3, 7, 9 এবং 11 নং স্থ রয়েছে। এখন এ সংখ্যাগুলোকে বাইনারিতে রূপান্তর করে যোগ করলে নির্ণেয় হামিং কোডগুলো পাওয়া যাবে। তবে যোগ সময় হাতের অঙ্ককে বাদ দিতে হবে।

এখানে,
$$\begin{array}{r} 3 = 0\,0\,1\,1 \\ 7 = 0\,1\,1\,1 \\ 9 = 1\,0\,0\,1 \\ 11 = 1\,0\,1\,1 \\ \hline 0\,1\,1\,0 \\ \downarrow\downarrow\downarrow\downarrow \\ P_4\,P_3\,P_2\,P_1 \end{array}$$

অর্থাৎ $P_1 = 0, P_2 = 1, P_3 = 1$ এবং $P_4 = 0$

এ বিটগুলো P এর স্থলে বসিয়ে হামিং কোড পাওয়া যাবে। সুতরাং নির্ণেয় হামিং কোড–

1	0	1	0	1	0	0	1	1	1	0	→বিট পজিশন
11	10	9	8	7	6	5	4	3	2	1	→হ্যামিং কোড

রিসিভারে এ কোডের যে-সব স্থানে 1 আছে যে-সব স্থানের সংখ্যা মানকে বাইনারিতে রূপান্তর করে তা আবার যোগ পরীক্ষা করে দেখতে হবে যে, যোগফল 0 কি না। যদি যোগফল 0 হয় তবে বুঝা যায় যে, গৃহীত ডাটাতে কোনো ভুল নেই, ভুল আছে। উক্ত হামিং কোডের 2, 3, 4, 7, 9 এবং 11নং স্থানে 1 রয়েছে।

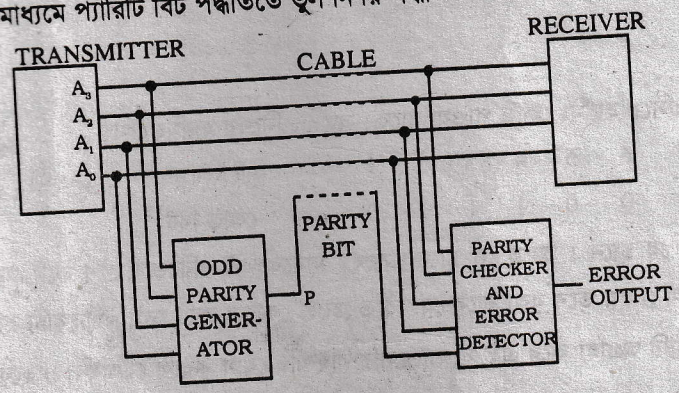
[illegible] এবং প্যারিটি চেকড কোডের দ্বারা সংশোধন (Describe the error detection and [illegible] with parity checked code) ঃ

[illegible] তথ্য স্থানান্তরণের সময় প্রবাহ পথে অনভিপ্রেত বৈদ্যুতিক শোরের (Noise) প্রভাবে প্রবাহিত তথ্যে [illegible] না [illegible] প্রেরিত 0 গন্তব্য স্থানে 1 হিসেবে আসতে পারে অথবা প্রেরিত 1 গন্তব্য স্থানে 0 হিসেবে আসতে পারে। [illegible] জটিল ডিজিটাল পদ্ধতিতে লক্ষ লক্ষ বিটের স্থানান্তর হয়ে থাকে। সুতরাং নির্ভরযোগ্যভাবে কাজ করার জন্য [illegible] একটি গুরুত্বপূর্ণ বিষয়। এক বিট ভুল নির্ণয় ও ভুল সংশোধনের জন্য সাধারণভাবে ব্যবহৃত পদ্ধতি হলো– [illegible] বিট পদ্ধতি, ২। হ্যামিং কোড পদ্ধতি।

[illegible] বিট পদ্ধতি ঃ এ পদ্ধতির সাহায্যে প্রেরিত তথ্যে এক বিট ভুল শনাক্ত করা যায় কিন্তু ভুল বিটের অবস্থান নির্ণয় করা [illegible] ভুল সংশোধন করা যায় না। এ পদ্ধতিতে তথ্যের সাথে একটি অতিরিক্ত বিট যুক্ত করে দেওয়া হয়। এ অতিরিক্ত [illegible] বিট বলা হয়।

[illegible] প্যারিটিবিট ছাড়া তথ্য 1001 প্যারিটি বিটসহ তথ্য P1001

[illegible] চিত্র ব্লক চিত্রের মাধ্যমে প্যারিটি বিট পদ্ধতিতে ভুল নির্ণয় পদ্ধতি আলোচনা করা হলো–



চিত্র ঃ ২.১ প্যারিটি বিটের মাধ্যমে ভুল নির্ণয়

[illegible] চারবিটের কোনো ডাটা এক স্থান হতে অন্য স্থানে পাঠাতে হবে। এ ডাটার ভুল নির্ণয় করতে হবে। [illegible] হতে চার বিট ডাটা একই সাথে রিসিভার ও প্যারিটি বিট জেনারেটরে পাঠাতে হবে। এ জেনারেটরের মাধ্যমে [illegible] বিট তৈরি করা হয়। রিসিভিং প্রান্তে পাঁচ (5) বিটের ডাটা (প্যারিটি বিটসহ)-কে প্যারিটি চেকার ও ডিটেক্টরে প্রদান করা [illegible] ডিটেক্টরটি নির্দিষ্ট প্যারিটি চেক করে। এক্ষেত্রে জেনারেটরটি যদি জোড় প্যারিটির জন্য হয়, তবে চেকারে জোড় প্যারিটি [illegible] এর আউটপুটের সংকেত ঘণ্টাটি বাজে না, অর্থাৎ ভুল নেই বুঝায়। আর চেকারে বিজোড় প্যারিটি পৌঁছলে এর সংকেত ঘণ্টা [illegible] উঠে, অর্থাৎ ভুল আছে বুঝায়। প্যারিটি জেনারেটর নির্ধারণের মাধ্যমে এটাকে জোড় এবং বিজোড় প্যারিটির জন্য কাজ [illegible] হয়। এভাবে যদি জোড় প্যারিটির জন্য হয়, তবে প্যারিটি বিটসহ 1 এর সংখ্যা জোড় না হলে ডিটেক্টরের সংকেত ঘণ্টা [illegible] আর বিজোড় প্যারিটির জন্য হলে প্যারিটি বিটসহ মোট 1 এর সংখ্যা বিজোড় না হলে ডিটেক্টরের ঘণ্টা বেড়ে উঠে। এভাবে [illegible] বিটের সাহায্যে ভুল শনাক্ত করা যায়।

[illegible] বিটসহ তথ্য প্রেরণ করা হয়। প্যারিটি জেনারেটরের সাহায্যে প্যারিটি বিট তৈরি করা হয়। গ্রাহক প্রান্তে গ্রহণকৃত [illegible] প্যারিটি চেক করা হয় অর্থাৎ 1 এর সংখ্যা চেক করা হয়। প্যারিটি বিটসহ প্রেরিত 1 এর সংখ্যা ও গৃহীত 1 এর সংখ্যা একই [illegible] তথ্যে ভুল নাই ধরা হয় আর ভিন্ন ভিন্ন হলে ভুল আছে ধরা হয়। তথ্য ট্রান্সমিশন পদ্ধতি জোড় প্যারিটি বা বিজোড় প্যারিটি হতে [illegible] জোড় প্যারিটি সিস্টেমের ক্ষেত্রে প্যারিটি জেনারেটর ও প্যারিটি চেকার উভয়ই জোড় হতে হয়।

২। হ্যামিং কোড পদ্ধতিতে ভুল নির্ণয় ও ভুল সংশোধন ঃ প্যারিটি চেকড কোডের মাধ্যমে ভুল নির্ণয় ও সংশোধন করা যায়। [illegible] প্রকার প্যারিটি চেকড কোড ব্যবহার করা হয়। হ্যামিং কোড এগুলোর মধ্যে উল্লেখযোগ্য। এখানে হ্যামিং কোডের মাধ্যমে এক [illegible] বিট ভুল নির্ণয় সংশোধন আলোচনা করা হলো। এই পদ্ধতিতে n বিট ডাটার সাথে k সংখ্যক প্যারিটি চেক বিট যুক্ত করে (n + k) [illegible] বিটের হ্যামিং কোড তথ্য তৈরি করা হয়। প্যারিটি চেক বিটের অবস্থান 2 এর পূর্ণ ঘাত দ্বারা নির্ধারণ করা হয়, অর্থাৎ 1, 2, 4, 8 [illegible] ইত্যাদি অবস্থানে ঢুকানো হয়। অবশিষ্ট অবস্থানে তথ্যের বিট প্রবেশ করানো হয়।

প্যারিটি চেক বিটের সংখ্যা k এর মান $2^k \geq n + k + 1$ সম্পর্ক দ্বারা নির্ধারণ করা হয়। একটি উদাহরণের সাহায্যে পদ্ধতিতে ভুল নির্ণয় ও সংশোধন বর্ণনা করা হলে, Y এর ASCII কোড 1011001; এক্ষেত্রে প্যারিটি চেক বিটের সংখ্যা কোড বিট হবে (7 + 4) = 11. 4টি প্যারিটি বিট 1, 2, 4 এবং 8 নং অবস্থানে বসিয়ে হ্যামিং কোড পাওয়া যাবে।

11	10	9	8	7	6	5	4	3	2	1	←	বিট অবস্থান
1	0	1	h_4	1	0	0	h_3	1	h_2	h_1	←	হ্যামিং কোড

প্যারিটি চেক বিটের মান নির্ণয় করার জন্য কোডের যে যে অবস্থানে 1 আছে তা শনাক্ত করা হয়। এক্ষেত্রে 3, 7, 9 নং স্থানে 1 রয়েছে। এখন এই সংখ্যাগুলোকে বাইনারিতে রূপান্তর করে যোগ করলে নির্ণেয় প্যারিটি বিটগুলোর মান পাওয়া যোগ করার সময় ক্যারি অগ্রাহ্য করতে হবে।

```
 3 = 0011
 7 = 0111
 9 = 1001
11 = 1011
-----------
     0110
     h4h3h2h1
```

এ বিটগুলো h এর স্থানে বসিয়ে হ্যামিং কোড পাওয়া যাবে। সুতরাং নির্ণেয় হ্যামিং কোড

11	10	9	8	7	6	5	4	3	2	1	←	বিট অবস্থান
1	0	1	0	1	0	0	1	1	1	0	←	কোড বিট

গ্রাহক যন্ত্রে এ কোডের যে যে স্থানে 1 আছে সে স্থানের সংখ্যা মানকে বাইনারিতে প্রকাশ করে যোগ করে পরীক্ষা করে যোগফল 0 কি না। যদি যোগফল 0 হয় তবে বুঝা যাবে যে গৃহীত তথ্যে কোনো ত্রুটি নেই। আলোচ্য হ্যামিং কোডটির 2, 3, 4 এবং 11 নং স্থানে 1 রয়েছে। ক্যারি অগ্রাহ্য করে এই সংখ্যাগুলোর বাইনারি যোগ করলে যোগফল 0 হবে। যেমন–

```
 2 = 0010
 3 = 0011
 4 = 0100
 7 = 0111
 9 = 1001
11 = 1011
-----------
     0000
```

যদি কোনো কারণে গৃহীত কোড ত্রুটিযুক্ত হয় তবে 1 এর স্থানগুলোর বাইনারি যোগফল 0 হবে না।

মনে করি, ত্রুটির কারণে 5 নং স্থানে 1 হয়ে গেছে; গ্রাহক যন্ত্রে 1 স্থানগুলোর ক্যারি অগ্রাহ্য করে বাইনারি যোগফল হবে–

```
 2 = 0010
 3 = 0011
 4 = 0100
 5 = 0101
 7 = 0111
 9 = 1001
11 = 1011
-----------
     0101
```

যোগফল 5 এবং এটি নির্দেশ করে যে 5 নং স্থানে গৃহীত বিট সঠিক নয়। উক্ত বিটটি উল্টিয়ে দিলে গৃহীত কোড সংশোধিত হবে।