CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

CHƯƠNG 2
CÁC KIỂU DỮ LIỆU VÀ THAO TÁC

1. **KIỂU DỮ LIỆU SỐ NGUYÊN**
2. **SỐ NGUYÊN BÙ 2**
3. **PHÉP TOÁN SỐ HỌC TRÊN BIT**
4. **PHÉP TOÁN LUẬN LÝ TRÊN BIT**
5. **KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG**

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.1 KIỂU DỮ LIỆU SỐ NGUYÊN

2.1.1 Số nguyên không dấu (unsigned integer)

Dùng để biểu diễn số lần lặp lại một tác vụ nhất định, hay chỉ địa chỉ của các ô nhớ.

Ví dụ: 102, 101101B

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.1 KIỂU DỮ LIỆU SỐ NGUYÊN

2.1.2 Số nguyên có dấu (signed integer)

• Dạng biểu diễn số âm dùng bit dấu và trị tuyệt đối, bit có trọng số cao nhất sẽ quy định dấu cho số có trị tuyệt đối ngay sau, nếu bằng 0 → số dương, 1 → âm.

• Dạng bù 1 sẽ biểu diễn số âm bằng việc đảo các trạng thái bit của số dương tương ứng, đảo từ 1 qua 0, và ngược lại.

• Dạng bù 2 sẽ biểu diễn số âm bằng dạng bù 1 của nó công thêm 1.

Dạng biểu diễn	Trị được biểu diễn		
	Trị tuyệt đối có dấu	Bù 1	Bù 2
00000	0	0	0
00001	1	1	1
00010	2	2	2
00011	3	3	3
00100	4	4	4
00101	5	5	5
00110	6	6	6
00111	7	7	7
01000	8	8	8
01001	9	9	9
01010	10	10	10
01011	11	11	11
01100	12	12	12
01101	13	13	13
01110	14	14	14
01111	15	15	15
10000	-0	-15	-16
10001	-1	-14	-15
10010	-2	-13	-14
10011	-3	-12	-13
10100	-4	-11	-12
10101	-5	-10	-11
10110	-6	-9	-10
10111	-7	-8	-9
11000	-8	-7	-8
11001	-9	-6	-7
11010	-10	-5	-6
11011	-11	-4	-5
11100	-12	-3	-4
11101	-13	-2	-3
11110	-14	-1	-2
11111	-15	-0	-1

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.2 SỐ NGUYÊN BÙ 2

Có hai bước trong quy luật tạo số bù 2 của một số:

- Lật ngược trạng thái bit biểu diễn từ 1 qua 0, từ 0 qua 1 trong mẫu, còn gọi là phép bù 1.

- Cộng 1 vào mẫu kết quả ở bước 1, để có mẫu kết quả sau cùng.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.2 SỐ NGUYÊN BÙ 2

Thí dụ 2.1: Tìm dạng bù 2 cho số -12

Mẫu nhị phân của trị tuyệt đối của toán hạng 12 là 01100.

Ta thực hiện hai bước như sau:

B1. Tìm bù 1 của 01100: 10011

B2. Cộng 1 vào dạng bù 1: 10100

$$\begin{array}{rr} & 01100 \\ + & 10100 \\ \hline & 100000 \end{array}$$

Kết quả là 0

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.3 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN SỐ HỌC

2.3.1 Cộng và trừ

Ví dụ 2.2: Tính biểu thức 11+3.Ta có:

Trị thập phân 11 được biểu diễn dưới dạng 01011

Trị thập phân 3 được biểu diễn ở dạng 00011

Tổng, có trị 14, là 01110

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.3 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN SỐ HỌC

2.3.1 Cộng và trừ

Thí dụ 2.3: Mô phỏng thực hiện phép trừ ở thao tác cộng ở ALU, tính biểu thức: 12 – 19.

Trước tiên, CPU phân tích để tính biểu thức trên ở dạng: 12 + (-19), sau đó tính bù 2 của 19 (010011) để có -19 (101101). Cộng 12, (001100), với -19 (101101):

```
  001100
+ 101101
--------
  111001
```

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.3 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN SỐ HỌC

2.3.1 Cộng và trừ

Thí dụ 2.4: Cộng một số với chính nó ($x + x$), tính 6 + 6.

Giả sử ta xét các mẫu có chiều dài 5 bit.

Mẫu nhị phân 5 bit của 6 là 00110, tức dạng khai triển là $0.2^4 + 0.2^3 + 1.2^2 + 1.2^1 + 0.2^0$

Khi ta thực hiện 6 + 6, hay 2.6, biểu thức khai triển sẽ là $0.2^5 + 0.2^4 + 1.2^3 + 1.2^2 + 0.2^1$

Ta có kết quả: 01100, tức dịch toán hạng ban đầu từng bit sang trái một vị trí.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.3 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN SỐ HỌC

2.3.2 Mở rộng dấu

Thao tác mở rộng thêm bit dấu (0 với số dương và 1 với số âm) vào phía trước dạng bù 2 sẽ không làm thay đổi giá trị của số ban đầu. Thao tác này được gọi là thao tác mở rộng dấu (Sign-EXTension), và thường được viết tắt là SEXT.

Ví dụ: 000101 -> 0000000000000101

100101 -> 1111111111100101

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.3 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN SỐ HỌC

2.3.3 Tràn số

Ví dụ: với chiều dài toán hạng là 5 bit, tính biểu thức 9 + 11, ta có:

```
    01001
+   01011
    10100
```

Kết quả ai cũng biết là 20, nhưng ta lại nhận được một số âm, do bit trọng số lớn nhất là 1, tức -12!

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

Một cách tổng quát, khi đề cập tới trạng thái luận lý *đúng*, thì ta có thể nghĩ ngay nó là bit 1, và ngược lại; còn nếu gặp trạng thái luận lý *sai*, thì cũng có nghĩa là ta có bit 0.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.1 Phép toán AND

AND là một hàm luận lý nhị phân, nó đòi hỏi hai toán hạng nhập, mỗi toán hạng là một trị luận lý 0 hoặc 1. Ta có thể hình dung toán hạng này hoạt động theo kiểu: cả hai đúng thì nó mới đúng.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.1 Phép toán AND

A	B	AND
0	0	0
0	1	0
1	0	0
1	1	1

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.1 Phép toán AND

Toán hạng này có thể tổng quát cho các mẫu *n* bit.Ví dụ 2.5:

Nếu *c* là kết quả AND của *a* và *b*, với *a* = 0011 1101 và *b* = 0100 0001, thì *c* bằng bao nhiêu ?

a : 0011 1101

b : 0100 0001

c : 0000 0001

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.1 Phép toán AND

Ví dụ 2.6:

Giả sử chúng ta có một mẫu nhị phân 8 bit được gọi là *A*, trong đó hai bit trọng số nhỏ nhất bên phải của *A* có ý nghĩa quan trọng. Làm sao cách ly hai bit này để xét ?

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.1 Phép toán AND

Chúng ta dùng *mặt nạ bit*.

Một *mặt nạ bit* là một mẫu nhị phân mà có thể làm cho ta thấy được hai phần khác nhau trong các bit của *A*, phần ta cần quan tâm và phần ta muốn bỏ qua.

Trong trường hợp này, mặt nạ bit 0000 0011 khi được AND với *A* sẽ tạo ra các bit 0 trong các bit từ vị trí 7 tới vị trí 2, còn các bit ở vị trí 1 và 0 thì sẽ được giữ nguyên.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.2 Phép toán OR

OR cũng là một phép toán luận lý nhị phân. Nó yêu cầu hai toán hạng đầu vào là hai trị luận lý. Khác với AND, chỉ cần một trong hai toán hạng đầu vào là 1 thì kết quả đầu ra của OR đã là 1.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.2 Phép toán OR

A	B	OR
0	0	0
0	1	1
1	0	1
1	1	1

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.2 Phép toán OR

Ví dụ 2.7:

Nếu c là kết quả OR của a và b,
với a = 0011 1101 và b = 0100 0001, thì c bằng bao nhiêu ?

a : 0011 1101
b : 0100 0001
c : 0111 1101

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.2 Phép toán OR

Ví du 2.8: Với một trạng thái bit đã có, ta muốn hai bit trọng số nhỏ nhất của nó phải có trạng thái xác định là 11, thì mặt nạ 0000 0011 sẽ được OR với trạng thái bit đã có. Chẳng hạn như:

0011 1101
0000 0011
0011 1111

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.3 Phép toán NOT

NOT là một hàm luận lý đơn toán hạng, nó chỉ cần một toán hạng nhập. Toán hạng này còn được gọi là toán hạng *bù*, vì nó thực hiện thao tác lật ngược trạng thái luận lý từ 1 qua 0, hoặc từ 0 qua 1.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.3 Phép toán NOT

A	NOT
0	1
1	0

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.3 Phép toán NOT

a: 0100 0001

thì c = NOT a: 10111110

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.4 Phép toán Exclusive-OR (EX-OR)

Phép toán này còn được gọi ngắn gọn là XOR. Đây là toán tử hai toán hạng. Đầu ra của XOR sẽ là 1 nếu hai đầu vào là khác nhau.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.4 Phép toán Exclusive-OR (EX-OR)

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.4 PHÉP TOÁN TRÊN BIT – PHÉP TOÁN LUẬN LÝ

2.4.4 Phép toán Exclusive-OR (EX-OR)

Ví dụ 2.9: Nếu *c* là kết quả XOR của *a* và *b*, với *a* = 0011 1101 và *b* = 0100 0001, thì *c* bằng bao nhiêu ?

a : 0011 1101
b : 0100 0001
c : 0111 1100

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

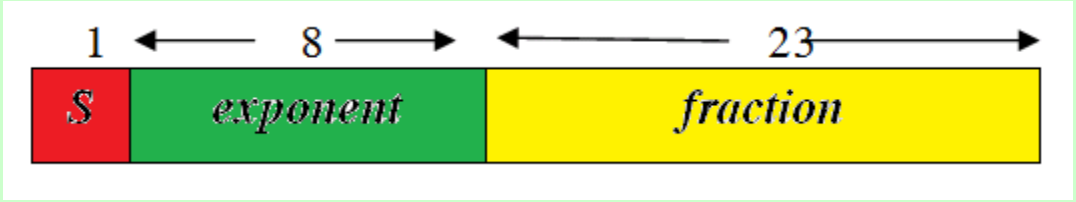
Kiểu dữ liệu dấu chấm động là cách giải quyết cho vấn đề biểu diễn số thập phân thay vì dùng dấu chấm tĩnh. Các kiến trúc tập lệnh (ISA) đều có kiểu dữ liệu dấu chấm động theo định dạng chuẩn IEEE 754.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

Một trong chúng là kiểu *float*, chiều dài 32 bit, có cấu trúc như sau:

1 bit cho dấu (dương hay âm)
8 bit cho tầm (vùng số mũ-exponent)
23 bit cho độ chính xác (fraction)



$N = (-1)^S \times 1.\text{fraction} \times 2^{\text{exponent}-127}$, $1 \leq \text{exponent} \leq 254$

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

Phần mũ dài 8 bit nhị phân, biểu diễn 256 trị không dấu, nhưng ta chỉ sử dụng 254 trị trong đó mà thôi. Vùng mũ chứa 0000 0000 (tức 0), hay 1111 1111 (tức 255) sẽ cho một ý nghĩa đặc biệt khác mà ta sẽ xét sau.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

Ví dụ 2.10: Hãy biểu diễn số $-6\frac{5}{8}$ ở dạng kiểu dữ liệu dấu chấm động.

Ví dụ 2.11: Hãy tìm trị cho dạng biểu diễn thuộc kiểu dấu chấm động sau:

0 01111011000000000000000000000000

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

Nếu phần mũ chứa 00000000 thì số mũ sẽ được xem la ` -126, phần trị mặc nhiên bắt đầu bằng bit 0 bên trái dấu chấm nhị phân, tới dấu chấm nhị phân, và theo sau là 23 bit phần trị bình thường, cụ thể

$$(-1)^{S} \times 0.fraction \times 2^{-126}$$

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

Ví dụ, dạng biểu diễn dấu chấm động

0 00000000 00001000000000000000000

có bit dấu bằng 0, nên là số dương, tám bit kế bằng 0, nên số mũ là -126, 23 bit cuối tạo ra dạng số 0.00001000000000000000000, tức bằng 2^{-5}. Như vậy, số được biểu diễn là $2^{-5} \cdot 2^{-126}$, tức 2^{-131}.

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

Thí dụ 2.12: Kiểm chứng trị kiểu dấu chấm động của các mẫu sau:

0 10000011 00101000000000000000000 là $1.00101 \times 2^4 = 18.5$

1 10000010 00101000000000000000000 là $-1 \times 1.00101 \times 2^3 = -9.25$

0 11111110 11111111111111111111111 là $1.111...11 \times 2^{127} \sim 2^{128}$

1 00000000 00000000000000000000001 là -2^{-149}

0 00000000 00000000000000000000000 là 0^+

1 00000000 00000000000000000000000 là 0^-

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

Nếu phần mũ chứa 11111111 thì ta sẽ có hai khả năng xảy ra:

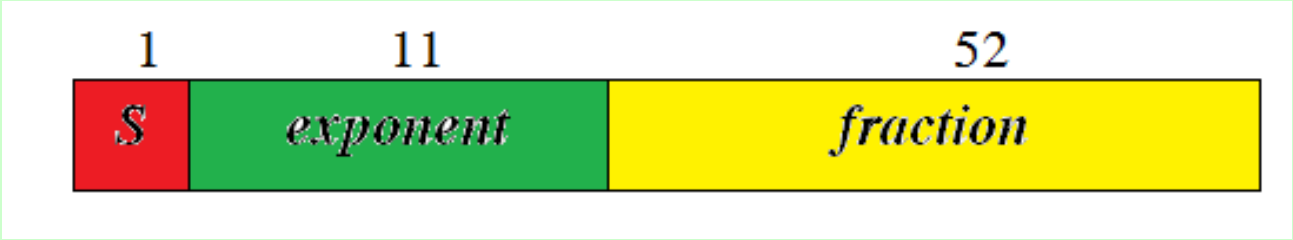
- Nếu phần trị bằng 0, số sẽ là dương vô cực (+∞) hay âm vô cực (-∞) tùy vào bit dấu.

-Nếu phần trị khác 0, lúc này việc biểu diễn số dấu chấm động sẽ không là một số (Not a Number - NaN), không quan tâm tới bit dấu. Dạng NaN này báo hiệu những thao tác không hợp lệ như nhân zero (0) với vô cực (∞).

CHƯƠNG 2
CÁC DỮ LIỆU VÀ THAO TÁC

2.5 KIỂU DỮ LIỆU DẤU CHẤM ĐỘNG (Floating point data type)

Tương tự, kiểu *double* có chiều dài 64 bit theo định dạng sau:



$N = (-1)^S \times 1.fraction \times 2^{exponent-1023}, \ 1 \le exponent \le 2046$