Toán học số 16 bit

TS Nguyễn Hồng Quang

Cộng hai số 16 bít

	100's	10's	1's
.	1	5	6
+	2	4	8
=	**4**	**0**	**4**

	256's	1's
.	1A	44
+	22	DB
=	**3D**	**1F**

- 65535 + 65535 = 131070 – số 17bit
- 44 + DB = 11F
- 1A + 22 = 3D

Thuật toán

	65536's	256's	1's
-		R6	R7
+		R4	R5
=	**R1**	**R2**	**R3**

- Số thứ nhất: Byte cao R6 byte thấp là R7
- Số thứ 2: byte cao là R4 và byte thấp là R5)
- Kết quả lưu trong R1, R2, và R3.
- R3 chỉ lưu trữ một bit

Các bước chương trình

```
MOV A,R7  ;Move the low-byte into the accumulator
ADD A,R5  ;Add the second low-byte to the accumulator
MOV R3,A  ;Move the answer to the low-byte of the result
```

```
MOV A,R6  ;Move the high-byte into the accumulator
ADDC A,R4 ;Add the second high-byte to the accumulator, plus
MOV R2,A  ;Move the answer to the high-byte of the result
```

```
MOV A,#00h  ;By default, the highest byte will be zero.
ADDC A,#00h ;Add zero, plus carry from step 2.
MOV R1,A    ;Move the answer to the highest byte of  the
```

Ví dụ sử dụng

```
;Load the first value into R6 and R7
MOV R6,#1Ah
MOV R7,#44h

;Load the second value into R4 and R5
MOV R4,#22h
MOV R5,#0DBh

;Call the 16-bit addition routine
LCALL ADD16_16
```

Phép trừ 16 bit

	256's	1's
.	22	DB
-	1A	F9
=	**07**	**E2**

	256's	1's
.	R6	R7
-	R4	R5
=	**R2**	**R3**

Hàm trừ 16 bit

```
SUBB16_16:
  ;Step 1 of the process
  MOV A,R7  ;Move the low-byte into the accumulator
  CLR C     ;Always clear carry before first subtraction
  SUBB A,R5 ;Subtract the second low-byte from the accumulator
  MOV R3,A  ;Move the answer to the low-byte of the result

  ;Step 2 of the process
  MOV A,R6  ;Move the high-byte into the accumulator
  SUBB A,R4 ;Subtract the second high-byte from the accumulator
  MOV R2,A  ;Move the answer to the low-byte of the result

  ;Return - answer now resides in R2, and R3.
  RET
```

Cách dùng hàm trừ

```
;Load the first value into R6 and R7
MOV R6,#22h
MOV R7,#0DBh

;Load the second value into R4 and R5
MOV R4,#1Ah
MOV R5,#0F9h

;Call the 16-bit subtraction routine
LCALL SUBB16_16
```

Phép nhân 2 số 16 bit

- Nhân 2 số 16 bit tạo ra số 32 bit
- Ví dụ nhân 2 số:
  - **25,136** x **17,198**. = 432,288,928
  - **6230h** x **432Eh** = 19,C4,32,48h

Ví dụ

	Byte 4	Byte 3	Byte 2	Byte 1
			62	**30**
*			**43**	**2E**
=			08	A0
		11	9C	
		0C	90	
	19	A6		
=	**19**	**C4**	**34**	**A0**

	Byte 4	Byte 3	Byte 2	Byte 1
*			R6	R7
*			R4	R5
=	**R0**	**R1**	**R2**	**R3**

Thuật toán

- Nhân R5 và R7, kết quả 16-bit lưu trong R2 vàR3.
- Nhân R5 và R6, cộng kết quả 16-bit vào R1 và R2.
- Nhân R4 và R7, cộng kết quả 16-bit vào R1 và R2.
- Nhân R4 và R6, cộng kết quả 16-bit vào R0 và R1

Kết quả

```
MOV A,R5 ;Move the R5 into the Accumulator
MOV B,R7 ;Move R7 into B
MUL AB   ;Multiply the two values
MOV R2,B ;Move B (the high-byte) into R2
MOV R3,A ;Move A (the low-byte) into R3
```

```
MOV A,R5    ;Move R5 back into the Accumulator
MOV B,R6    ;Move R6 into B
MUL AB      ;Multiply the two values
ADD A,R2    ;Add the low-byte into the value already in R2
MOV R2,A    ;Move the resulting value back into R2
MOV A,B     ;Move the high-byte into the accumulator
ADDC A,#00h ;Add zero (plus the carry, if any)
MOV R1,A    ;Move the resulting answer into R1
MOV A,#00h  ;Load the accumulator with  zero
ADDC A,#00h ;Add zero (plus the carry, if any)
MOV R0,A    ;Move the resulting answer to R0.
```

Nhân 16 bít (tiếp)

```
MOV A,R4    ;Move R4 into the Accumulator
MOV B,R7    ;Move R7 into B
MUL AB      ;Multiply the two values
ADD A,R2    ;Add the low-byte into the value already in R2
MOV R2,A    ;Move the resulting value back into R2
MOV A,B     ;Move the high-byte into the accumulator
ADDC A,R1   ;Add the current value of R1 (plus any carry)
MOV R1,A    ;Move the resulting answer into R1.
MOV A,#00h  ;Load the accumulator with zero
ADDC A,R0   ;Add the current value of R0 (plus any carry)
MOV R0,A    ;Move the resulting answer to R1.
```

```
MOV A,R4   ;Move R4 back into the Accumulator
MOV B,R6   ;Move R6 into B
MUL AB     ;Multiply the two values
ADD A,R1   ;Add the low-byte into the value already in R1
MOV R1,A   ;Move the resulting value back into R1
MOV A,B    ;Move the high-byte into the accumulator
ADDC A,R0  ;Add it to the value already in R0 (plus any carry)
MOV R0,A   ;Move the resulting answer back to R0
```

Cách sử dụng

```
;Load the first value into R6 and R7
MOV R6,#62h
MOV R7,#30h

;Load the first value into R4 and R5
MOV R4,#43h
MOV R5,#2Eh

;Call the 16-bit subtraction routine
LCALL MUL16_16
```

Phép chia 16 bit

```
              5678 : 12 = ?
--------------------------
- 4 * 1200 = 4800
             ----
              878
- 7 *  120 =  840
              ---
               38
- 3 *   12 =   36
               --
                2
Result: 5678 : 12 = 473 Remainder 2
==================================
```

```
          1101
       ---------
1001 ) 1110101
         1001
         ----
          1011
          1001
          ----
            1001
            1001
            ----
```

Dịch trái

```
  MOV B,#00h ;Clear B since B will count the number of left-shifted
bits
div1:
  INC B      ;Increment counter for each left shift
  MOV A,R2   ;Move the current divisor low byte into the accumulator
  RLC A      ;Shift low-byte left, rotate through carry to apply highest
bit to high-byte
  MOV R2,A   ;Save the updated divisor low-byte
  MOV A,R3   ;Move the current divisor high byte into the accumulator
  RLC A      ;Shift high-byte left high, rotating in carry from low-byte
  MOV R3,A   ;Save the updated divisor high-byte
  JNC div1   ;Repeat until carry flag is set from high-byte
```

179 - R1/R0 00000000 10110011
8 - R3/R2 00000000 00001000

C/R1/R0 0 00000000 10110011
C/R3/R2 1 00000000 00000000

Phép trừ lặp lại

```
div2:         ;Shift right the divisor
  MOV A,R3    ;Move high-byte of divisor into accumulator
  RRC A       ;Rotate high-byte of divisor right and into carry
  MOV R3,A    ;Save updated value of high-byte of divisor
  MOV A,R2    ;Move low-byte of divisor into accumulator
  RRC A       ;Rotate low-byte of divisor right, with carry from high-
byte
  MOV R2,A    ;Save updated value of low-byte of divisor
  CLR C       ;Clear carry, we don't need it anymore
  MOV 07h,R1 ;Make a safe copy of the dividend high-byte
  MOV 06h,R0 ;Make a safe copy of the dividend low-byte
  MOV A,R0    ;Move low-byte of dividend into accumulator
  SUBB A,R2   ;Dividend - shifted divisor = result bit (no factor, only
0 or 1)
  MOV R0,A    ;Save updated dividend
  MOV A,R1    ;Move high-byte of dividend into accumulator
  SUBB A,R3   ;Subtract high-byte of divisor (all together 16-bit
substraction)
  MOV R1,A    ;Save updated high-byte back in high-byte of divisor
  JNC div3    ;If carry flag is NOT set, result is 1
  MOV R1,07h ;Otherwise result is 0, save copy of divisor to undo
subtraction
  MOV R0,06h
div3:
  CPL C       ;Invert carry, so it can be directly copied into result
  MOV A,R4
  RLC A       ;Shift carry flag into temporary result
  MOV R4,A
  MOV A,R5
  RLC A
  MOV R5,A
  DJNZ B,div2 ;Now count backwards and repeat until "B" is zero
```


Thuật toán khác

- Set quotient to zero
- Repeat while dividend is greater than or equal to divisor
  - Subtract divisor from dividend
  - Add 1 to quotient
- **End of repeat block**
- quotient is correct, dividend is remainder
- STOP

Dividend: 1011
Divisor: 10

START DEMO | NEXT STEP | RESET

Dividend	Quotient
1011 - 10	☒☒☒☒0
1001 - 10	☒☒☒☒1
111 - 10	☒☒☒10
101 - 10	☒☒☒11
11 - 10	☒☒100
1	☒☒101

Bài tập

- Khởi tạo kết quả = 0
- while
  - Nếu số bị chia còn lớn hơn hoặc bằng thương số
    - Trừ số bị chia cho thương số
    - Tăng kết quả lên 1
- Kết thúc:
  - Số bị chia còn lại là phần dư