# *Summary of Shift Instructions*



## Types of Shift Instructions

Shift instructions come in three basic varieties:

(1) Circular shift: Each bit moves in the direction of the shift, with the bit at the end moving around to replace the bit shifted out of the beginning bit position (Circular Right Shift: ***crsh***; Circular Left Shift: ***clsh***). NOTE that this instruction may come in either or both of two variants, circular shift WITH carry and circular shift WITHOUT carry, depending upon whether or not the bit in the Carry Flag participates in the shift cycle.

(2) Logical shift: Each bit moves in the direction of the shift, with the bit at the end replacing the Carry Flag, and a '0' replacing the bit shifted out of the beginning bit position (Logical Right Shift: ***lrsh***; Logical Left Shift: ***llsh***). NOTE that the bit that is shifted out of the register goes into the Carry Flag.

(3) Arithmetic shift: Arithmetic Shift Left is identical to Logical Shift Left, but the Arithmetic Shift Right differs from the Logical Shift Right in that instead of always shifting a '0' into the leftmost bit position, whatever bit currently occupies the leftmost bit position (whether a '0' or a '1') is replicated in that position in addition to being shifted to the second-to-the-left position (Arithmetic Right Shift: ***arsh***; Arithmetic Left Shift: ***alsh***) . NOTE that the bit that is shifted out of the register goes into the Carry Flag.

## Examples:

### *D7 (value = -41 in Two's-Complement) crsh ⇒ EB (value = -21)*

| D | | | | 7 | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

⇓

| E | | | | B | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |

In these examples, the Circular Shift takes place without the participation of the Carry Flag. Therefore, at the end of this and any other circular shift operation the Carry Flag is unchanged from whatever value it had held previously. In this case, the result happens to be that of a division by two with the answer rounded down, but this will not always be the case. See if you can think it through and figure out why.

### *D7 (value = -41 in Two's-Complement) clsh ⇒ AF (value = -81)*

| D | | | | 7 | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

⇓

| A | | | | F | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |

Once more, at the end of the operation the Carry Flag is unchanged from whatever value it had held previously. In this case, the result happens to be close to a multiplication by two, but that will not always be the case. Try again to think why.

### *D7 (value = -41 in Two's-Complement) alsh ⇒ AE (value = -82)*

| D | | | | 7 | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

⇓

| A | | | | E | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |

At the end of this operation, the Carry Flag will have a value of 1. In this case, the result is exactly equivalent to multiplication by two. Why will this always be so, barring an overflow?

### *D7 (value = -41 in Two's-Complement) llsh ⇒ AE (value = -82)*

| D | | | | 7 | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

⇓

| A | | | | E | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |

At the end of this operation, the Carry Flag will have a value of 1. Note that there is absolutely no difference between an arithmetic left shift and a logical left shift.

### *D7 (value = -41 in Two's-Complement) arsh ⇒ EB (value = -21)*

| D | | | | 7 | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

⇓

| E | | | | B | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |

Notice that the effect of an arithmetic right shift in Two's-Complement is to divide by 2, rounding downwards. At the end of this operation, the Carry Flag will have a value of 1.

### *D7 (value = -41 in Two's-Complement) lrsh ⇒ 6B (value = +107)*

| D | | | | 7 | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

⇓

| 6 | | | | B | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |

At the end of this operation, the Carry Flag will have a value of 1. Why is there no apparent relationship between the numeric values of the numbers contained in the register before and after the operation?

### *AE (value = -82 in Two's-Complement) lrsh ⇒57 (value = +87)*

| A | | | | E | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

⇓

| 5 | | | | 7 | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

At the end of this operation, the Carry Flag will have a value of 0. Again, there is no apparent relationship between the numeric values of the numbers contained in the register before and after the operation.

### *AE (value = -82 in Two's=Complement) arsh ⇒ D7 (value = -41)*

| A | | | | E | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

⇓

| D | | | | 7 | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |

Notice again that the effect of an arithmetic right shift in Two's-Complement is to divide by 2. At the end of this operation, the Carry Flag will have a value of 0.