DR R1,R2

Divide Register

RR

Op Code	R_1R_2

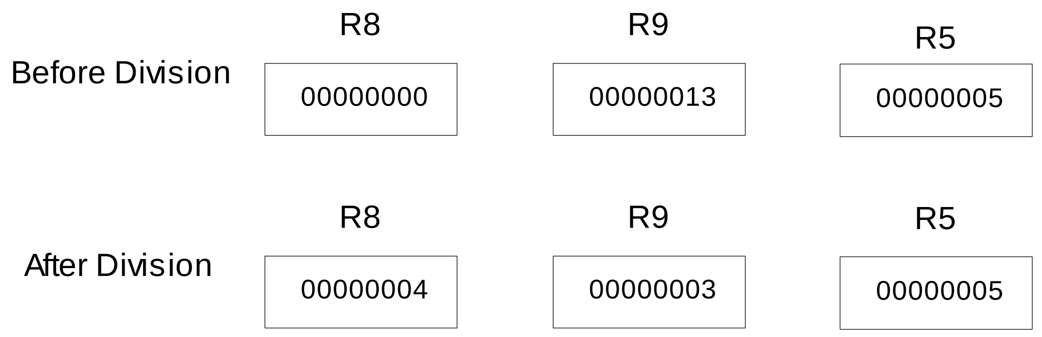
The Divide Register instruction performs 2’s complement binary integer division and returns a quotient and a remainder. Operand 1 names an even register of an “even-odd” consecutive register pair. For instance, R2 would be used to name the R2 / R3 even-odd register pair, and R8 would be used to name the R8 / R9 even-odd register pair. Operand 2 names a register that contains the divisor. Before the division, the even-odd register pair must be initialized with the dividend, which is effectively a 64-bit 2’s complement integer. After the division, the remainder is contained in the even register and the quotient is contained in the odd register. The sign of the quotient is determined by the laws of algebra, and the sign of the remainder is the same as the sign of the dividend.

	Even Register	Odd Register
Before Division	Dividend (left)	Dividend (right)
After Division	Remainder	Quotient

In preparing to divide a fullword, a common practice is to load the fullword in the even register and algebraically shift it to the odd register using **SRDA**. This practice propagates the appropriate sign bit throughout the even register. (If the dividend is positive, the even register is populated with binary 0’s. If the dividend is negative, the even register is populated with binary 1’s.) Here is an example where we compute A/B where A and B are fullwords.

```
     L    R8,A      PUT THE DIVIDEND IN THE EVEN REGISTER
     SRDA R8,32     ALGEBRAICALLY SHIFT R8 INTO R9
     L    R5,B      PUT THE DIVISOR IN R5
     DR   R8,R5     DIVIDE A BY B
     ...
A    DC   F’19’     DIVIDEND
B    DC   F’5’      DIVISOR
```

The diagram below illustrates the above division just after R8 has been shifted.



The diagram illustrates that the results of the integer division of 19 by 5 is a remainder of 4 in R8, and a quotient of 3 in R9.

Examples

Some Unrelated Divide Register Instructions

```
L    R6,=F’100’   DIVIDEND INITIALLY GOES IN THE EVEN REGISTER
SRDA R6,32        ... AND IS SHIFTED TO THE ODD REGISTER
L    R9,=F’10’    DIVISOR GOES IN R9
DR   R6,R9        ... BEFORE DIVIDING
                  R6 (REMAINDER) = X’00000000’,
                  R7 (QUOTIENT)  = X’0000000A’

L    R6,=F’100’   DIVIDEND INITIALLY GOES IN THE EVEN REGISTER
SRDA R6,32        ... AND IS SHIFTED TO THE ODD REGISTER
L    R4,=F’8’     DIVISOR GOES IN R4
DR   R6,R4        ... BEFORE DIVIDING
                  R6 (REMAINDER) = X’00000004’,
                  R7 (QUOTIENT)  = X’0000000C’

L    R6,=F’100’   DIVIDEND INITIALLY GOES IN THE EVEN REGISTER
SRDA R6,32        ... AND IS SHIFTED TO THE ODD REGISTER
SR   R5,R5        ZERO OUT R5
DR   R6,R5        ... BEFORE DIVIDING
                  ABEND - DIVISION BY 0 NOT ALLOWED

L    R6,=F’-100’   DIVIDEND INITIALLY GOES IN THE EVEN REGISTER
SRDA R6,32        ... AND IS SHIFTED TO THE ODD REGISTER
L    R10,=F’-8’    DIVISOR GOES IN R8
DR   R6,R10       ... BEFORE DIVIDING
                  R6 (REMAINDER) = X’FFFFFFFC’,
                  R7 (QUOTIENT)  = X’0000000C’
```

Tips

1) Know your data! If the divisor might be zero, you must protect your divisions by testing the divisor beforehand.

```
                  LTR   R5,R5           ASSUME DIVISOR IS IN R5
                  BZ    ZERODIV         BRANCH IF DIVISOR IS 0
                  DR    R8,R5           O.K. TO DIVIDE NOW
                  ...
         ZERODIV  EQU   *
                  (CODE TO HANDLE A ZERO DIVISOR)
```

2) Unlike the MR instruction, you must initialize the even register. The even/odd register pair must contain a 64 bit binary integer before you execute the instruction.