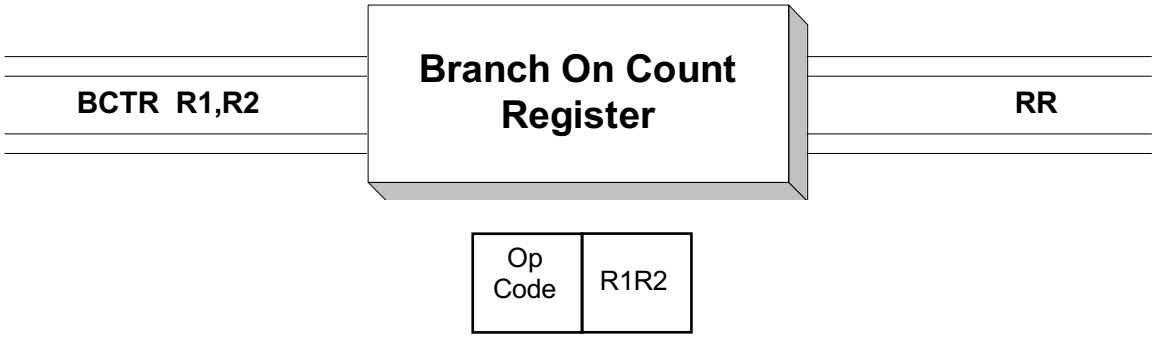


BCTR is the RR version of **BCT**. It is used to support counted loops - loops in which the number of iterations is known before entry to the loop body. Operand 1 is a register which contains a count - the number of iterations the loop should perform. Each time the **BCTR** is executed, the operand 1 register is decremented by 1. If the result is not zero, a branch is taken to the address stored in the register denoted by operand 2. If the result is zero, no branch is taken and execution continues with the instruction following the **BCTR**. Here is an example of how BCTR can be used to create a counted loop.

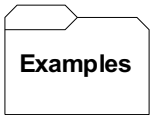
```
          LA    R10,LOOP  PUT TARGET ADDRESS IN R10
          LA    R8,3      LOOP 3 TIMES
LOOP      EQU   *
          ... (LOOP BODY GOES HERE)
          BCTR  R8,R10    BRANCH TO "LOOP" IF NOT ZERO
          MVC   X,Y
```

First the address of the loop body (the target address) is loaded into register 10. Then the number of iterations, 3, is put in R8. The loop body is executed. The **BCTR** decrements R8 to 2, and since the result is not zero, a branch is taken to the address in R10, which is LOOP. The loop body is executed again and the **BCTR** decrements R8 to 1. Since the result was not zero, another branch is taken to LOOP. The third execution of the loop body occurs and **BCTR** decrements R8 to 0. No branch is taken in this case, and execution falls through the loop and continues with the **MVC** instruction.

BCTR is sometimes coded with R0 specified in operand 2 to indicate that no branch should ever be taken. Execution always continues with the next instruction.

```
          BCTR   R5,R0
```

In this case, R5 is decremented by 1. Since R0 was coded as the second operand, execution continues with the instruction following the branch. Coding **BCTR** in this way provides an easy and efficient way to subtract 1 from a register.



Some Unrelated BCTR's

```
          R4 = X'00000008'
          R5 = A(HERE)
          R6 = X'00000001'
          R7 = X'00000002'
BCTR      R4,R5  R4 = X'00000007', BRANCH TO "HERE"
BCTR      R4,R0  R4 = X'00000007', DON'T BRANCH, FALL THROUGH
BCTR      R6,R5  R6 = X'00000000', DON'T BRANCH
BCTR      R7,R5  R7 = X'00000001', BRANCH TO "HERE"
```

Tips

1) BCTR is often used to decrement a register by 1 since it only affects the operand 1 register, and the value 1 doesn't have to be defined in memory.