Changes to Code:

Current Solution:

**Note: There is a logic error when you do one full negative rotation to 0, then rotate in the positive direction, then it resets to -359. Still need to fix this.**

**-359 => 0 => -1 => 0 => -359**

**It does work as intended when turning full rotation positively, then negatively, it would reset to -1 after 0.**

Commented out REG\_TC0\_RC0 = 360; found in void setup(). There is no need to reset counter on TC\_CV == RC, as modules will be used in loop function to control overflow of 360 and -360.

Completely changed void loop() function. I included two different ways we could approach overflow.

void loop() {

// 359 => 0 => 1 => 2 => 1 => 0 => -1 => -2

// We start counting from 0, since this is C. Once REG\_TC0\_CV0 or REG\_TC0\_QISR hits -360,

// we would be at 0 and the next number should be -1. If EG\_TC0\_CV0 or REG\_TC0\_QISR hits -360,

// we would be at 0, since -360 and 0 are equivalent. This should take care of any extreme cases

// of reaching max int like before.

int initial\_angle = REG\_TC0\_CV0;

int final\_angle = REG\_TC0\_QISR;

if (initial\_angle == -360 || final\_angle == -360 || initial\_angle == 360 || final\_angle == 360){

initial\_angle = 0;

final\_angle = 0;

Serial.println(initial\_angle);

Serial.print(" ");

Serial.println(final\_angle >> 8);

} else {

initial\_angle = initial\_angle % 361;

final\_angle = final\_angle % 361;

Serial.println(initial\_angle);

Serial.print(" ");

Serial.println(final\_angle>> 8);

}

}

Initial Solution:

**If you were to do something like this and did 2 full rotation negatively. Then you would have to do two full rotations to get back to positive. It is in the same vein if you do full 2 rotations to the left for example, to get back to negative you would have to do two full negative rotations.**

void loop() {

Serial.println(REG\_TC0\_CV0 % 360);

Serial.print(" ");

Serial.println(REG\_TC0\_QISR % 360>> 8);

}