Motor Speed Control

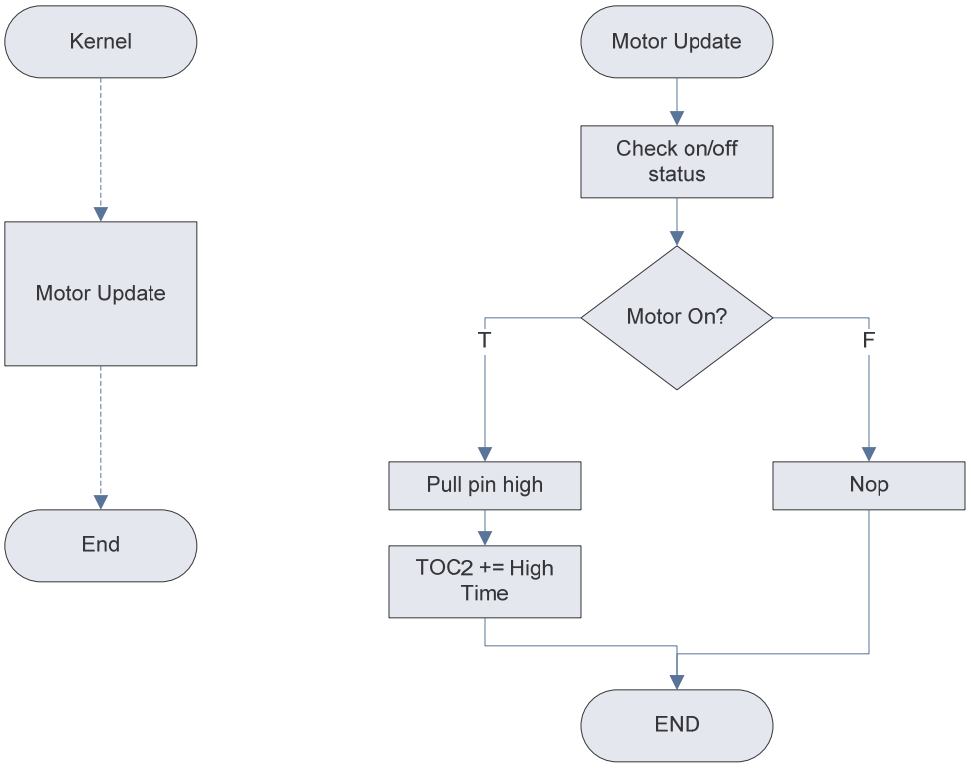
Speed is proportional to duty cycle. However, it is independent of period.

Using the Kernel

It is possible to integrate the motor speed control into the kernel of our operating system.

Assume that TOC has been initialized to pull a pin low.

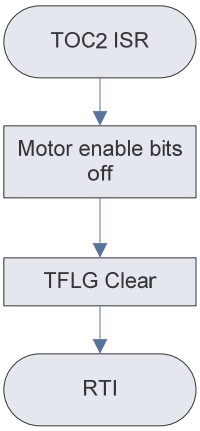
This adds minimal load to the kernel and eliminates the need for a second ISR.



This flow gets replicated for a second motor – one for left and one for right.

Because our motors are not connected to one of the TOC pins, we must have an ISR write values to them.

```
If the motor should be on,
        LDAA #____
        STAA    $7000
If the motor should be off,    ; This becomes unnecessary
        LDAA    #__OFF__
        STAA    $7000
TOC2 = TCNT + __HI TIME VALUE__
```



Consider using equates to form a lookup table and use that to test for which motor needs to be on.

Embedded systems seem very hard, but they are not. You are using a language to make bit movements that control the system. In our system, the bit movements allow the timer interrupt event to occur at specific times. Then, when the interrupt actually occurs, we want to do bit movements that will control the motors.

Also consider creating this separate from the OS, just to ensure it works, and then integrate it into the OS.