Synchronizing Execution to Real-Time by Polling TMR0

INTRODUCTION

Polling TMR0 to synchronize a process with real-time is a common technique. Usually, the timer is reset each time so it repeats a specified period, polling the timer with code like:

```
WAIT    BTFSS     TMR0,7
GOTO      WAIT
```

This method has a slightly unpredictable latency depending on which of the three instruction cycles within the loop the timer expires on. If the timer is reset by code like:

```
MOVLW     $80-delay
MOVWF     TMR0,F
```

The latency error accumulates. Replacing the previous reset code with:

```
MOVLW     $80-delay + 3
ADDWF     TMR0,F
```

avoids this cumulative drift of real-time synchronization. The +3 to run is required to make up for the latency in the ADDWF TMR0 instruction. So, the complete timer-synchronous wait loop is as below:

```
WAIT     BTFSS     TMR0,7
GOTO      WAIT
MOVLW     $80-delay + 3
ADDWF     TMR0,F
GOTO      WAIT
```