QUICKCODE™

Transducer Measurement

Author: *Zeljko Aleksic*
Faculty of Electrical Engineering, University of Belgrade
Belgrade, Yugoslavia
Email: zeljko@el.etf.bg.ac.yu

OVERVIEW

In many transducer types, quantitative changes in the input physical variable cause correspondent changes in output resistance or capacitance at the transducer electrical port, or in output signal directly (voltage or current). To convert those changes to digital values using the PIC12C5XX, the simplest and cheapest method is represented in measuring the frequency or the duration of the voltage ramp generated by charging and discharging external or transducer output capacitance. For faster measurements (to save energy, for example) ramp duration conversion is prefered. Conversion can be realized using the internal TMR0 timer, with internal clock (for lower cost). Conversion ends when the predetermined ramp voltage level is achieved (input pin change, provided by the analog comparator) or when timer overflow occures. If the charging rate can be chosen to be sufficiently high, measurement speed is limited by the speed of software testing for the occurrence of any of those two ending events. This problem is more complicated by the absence of interrupt capabilities of PIC12C5XXs.

Straightforward solution to this problem is represented in Example 1.

One cycle of this loop consumes five instruction cycles. Hence, minimal value of the prescaler modulus is eight, if count accuracy has to be preserved. QuickCode Idea given below represents the loop that consumes only four instruction cycles. In this case, minimal value of the prescaler modulus is four, and the maximal conversion speed is doubled.

APPLICATION OPERATION

Let GP5 turns on the power to the transducer, and is equal to '1' during the conversion. Let GP2 be controller input for the analog comparator output level, and let it be '0' during conversion. Let all other GPIO bits be '0'. Hence, reading the GPIO register during conversion gives the content equal to 0x20 (when analog comparator trips, the content is 0x24). Beginning of the loop has to be positioned at the program memory address equal to the result of the "movf GPIO,W" instruction during conversion. Timer overflow effectively resets the PIC, and input pin change branches to the program memory address equal to the result of the "movf GPIO,W" instruction after pin change (in case of the abovementioned content, 0x24, it effectively continuates program execution).

EXAMPLE 1: STRAIGHTFORWARD SOLUTION

```
WAIT:   incfsz   TMR0,W
        btfsc    GPIO,INPUT     ; INPUT is '0' during conversion
        goto     EOC
        goto     WAIT
EOC:    . . .
```

APPENDIX A: SOFTWARE LISTING:

```
                ORG             0x20
;
                incfsz          TMR0,W
                movf            GPIO,W
                movwf           PCL
                nop
```