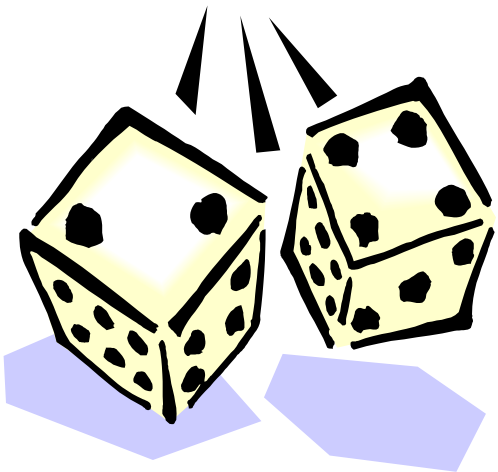
ELECTRONIC DICE

Something to play with for the PIC10F202/PIC10F206

by
Melanie Newman
12 March 2006
melanie@tekpumps.com

A simple demonstrator in **MeLabs PICBasic Pro** showing an example of controlling a heap of LEDs, a Switch Input and Auto-Power-OFF with only 3½ I/O's and a dozen or so components (of which eight are LEDs!).



1. What's an Electronic Dice?

What you've never played board games before? This is an electronic version that will give you a RANDOM Number between 1 and 6. There are seven LEDs arranged in the classic DICE layout which will illuminate for a period once you "Roll" the Dice by either pressing a Button, or, shaking the unit (assuming you've a tilt-switch fitted).

It's written 100% in MeLabs PICBasic Pro, with absolutely *NO* embedded Assembler. OK, so I lie, there is *ONE* line of Assembler... But I'm sure everyone will be able to follow that one!

2. Hardware

This project is designed around the 6-pin (or 8-pin PDIP) PIC10F202 or PIC10F206. We have around 340 or so words of code, which unfortunately just won't fit into the 256 words available in the bottom end PIC10F200 or PIC10F204. You can easily convert this to any other PIC in existance, but the challenge here was to do something useful with Microchips bottom-end baby product.

2.1 Hardware Description

Seven LED's are connected to three of the PIC's I/O's GP0, GP1 and GP2. Six of those LED's are connected parallel in pairs. D1 is on it's own, but D2 & D3 will light as a pair, D4 & D5 will light as a pair and D6 & D7 will light as a pair. This gives us four possible LED illuminations. Since each PIC I/O can be tri-stated, it is obvious that we can actually connect SIX unique LEDs to only three I/O's and turn any one of them ON or OFF. Here we only need to use FOUR out of the six possible combinations available to us. Current limiting is performed by the PIC and through software time-slicing which dispenses with the need for any current limiting Resistors.

Switch SW1 is connected to the INPUT ONLY pin of the PIC and serves as the "Spin" (or "Roll" the Dice) Button. LED D8 serves as a confidence indicator that we're "rolling", and R1 is a current limiter for D8, needed because we're connecting the LED directly across the Supply when that Button is pressed. We can additionally connect a tilt-switch so we can physically shake our electronic dice just like the real thing - only just don't throw it against the back of the craps table!

Those baby PIC10Fs run down to 2v, so we can simply connect a 3v coin cell to power our Dice. After a few seconds of display, the Dice will auto Power-OFF leaving us a remanent OFF current of between 5-10uA. You want to tether yours to a Wall Cube in case it gets lost? - go for it!

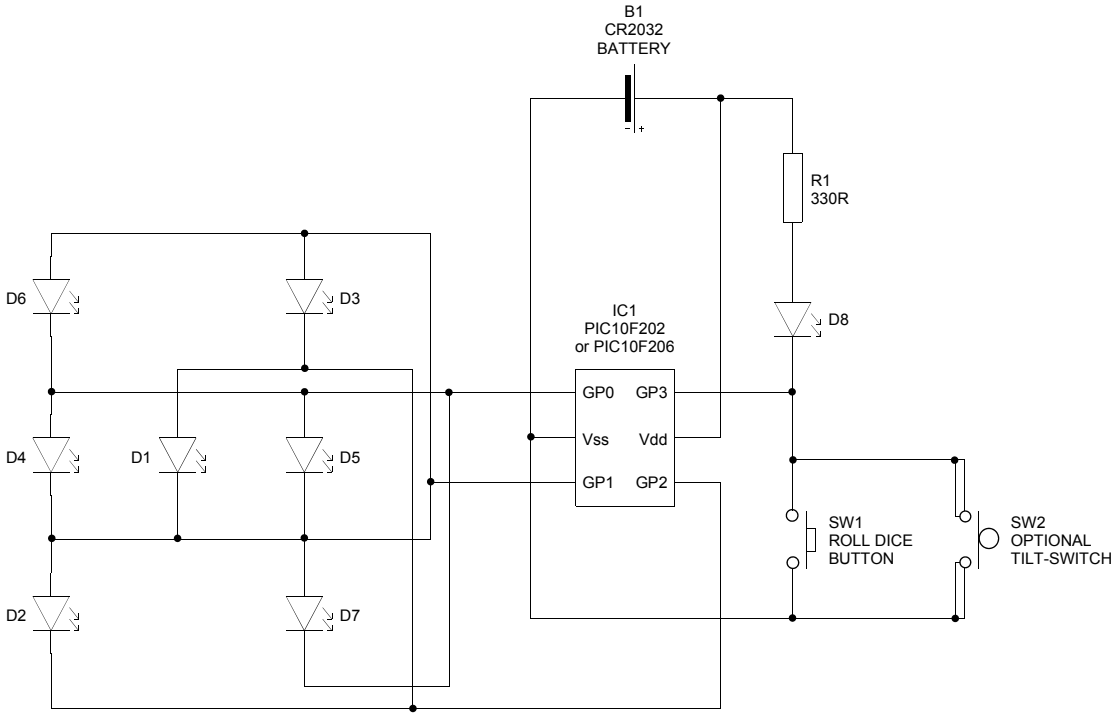


Figure 1. Melanie's Electronic Dice Schematic

3. User Operation

This is really complicated - so pay attention...

Press the **ROLL** Button (or shake the box if a tilt switch is fitted) to generate a Random Number. As long as your finger is on the Button, (or you keep shaking the box), the Dice keeps “rolling”.

Release the Button (or place the Dice back on the table) and a the generated Random Number (from 1 to 6) will be displayed for a few Seconds, after which time the unit will switch OFF waiting for the next “roll”.

4.0 Software Description

Pressing the Button does two things... Firstly it wakes the PIC from it’s low-power quiescent SLEEP, and secondly it increments a counter *RollCount*. As long as your finger is on the Button SW1 (or tilt Switch SW2 is closed), *RollCount* keeps incrementing. This is the purpose of the *While/Wend* loop. This incrementation of the variable *RollCount* is fast... Thousands of times per second. It is not possible to determine and stop the count on any given number, so as good as dammit, this is our Random Number generator. If you’re shaking the box, you tend to shake, stop, shake some more. Just as if when you press a Button you press, stop and maybe press a few more time in succession. Following the *While/Wend* loop is a small *For/Next* loop that just checks for a second or so that if you press the Button again, or keep shaking the box, we’ll go back to incrementing our counter.

Once you get bored with shaking the box or pressing the Button we get a final value in *RollCount*. From this value we calculate the remainder of a Divide-by-6 operation (obtaining a final value of between 0 and 5) which represents our 1-6 Dice Roll Result.

Finally we need to Display our Result. This really is the most complicated part of our program. The display LED’s are connected in a classic Dice arrangement, but due to the lack of I/O pins, only one LED (or LED pair) can be illuminated at any time. However, thanks to the limitations of the human eye, if we illuminate any LED for a short period (in our case 10mS) and then move onto another LED and do the same, it appears as if both are illuminated simultaneously. This also has a secondary effect of limiting the current flowing through an I/O pin to a short period which will not stress-out the I/O, thereby dispensing with the need for current limiting Resistors.

To display a DICE throw of **ONE**, we simply switch-on LED-A.

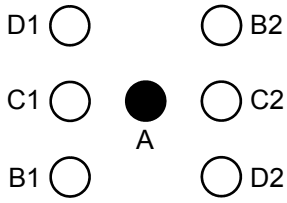


Figure 2a. DICE Number 1

For a DICE throw of **TWO** we switch-on the connected LED-B pair.

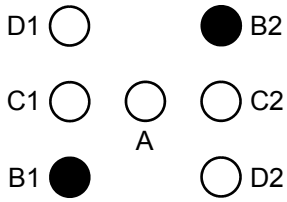


Figure 2b. DICE Number 2

For a DICE throw of **THREE**, we alternatively cycle between switching on LED-A for 10mS followed by the LED-B pair for 10mS.

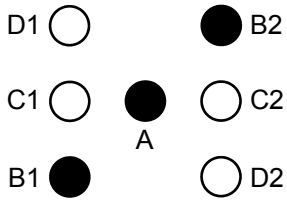


Figure 2c. DICE Number 3

To display a **FOUR**, we alternatively cycle between switching on the LED-B pair followed by the LED-D pair. A **FIVE** requires us to Display LED-A, followed by the LED-B pair followed by the LED-D pair. Finally a **SIX** cycles between the LED-B pair, the LED-C pair and the LED-D pair.

The FIVE and SIX pairs take 30mS to cycle (10mS per LED pair). THREE and FOUR take 20mS to cycle, but we pad-out the display operation with an extra 10mS delay. ONE and TWO take 10mS to cycle, and we pad those out by inserting an extra display and 10mS delay. In this manner we equalise the display time and also the brightness levels between the LEDs.

Once we have displayed the Result for a period of time, our program executes the ***@ SLEEP*** statement - the only Assembler instruction in our entire PICBasic program. This puts the PIC into low-power quiesce mode consuming somewhere between 5-10uA. The PIC will stay in sleep mode until woken by a press (or shake) starting a new "Roll". Naturally we have previously enabled the *wake-on-pin-change* in the PIC by setting the appropriate bits in the *STATUS* and *OPTION* registers, and immediately prior to executing the *@ SLEEP* we read the input pins to clear any pending event flags that might immediately wake our PIC.

5.0 Questions

If you have questions, commendations or condemnations, don't email or message me off-list, please keep them on the forum...

www.picbasic.co.uk/forum

... posting them under the topic thread in the *CODE EXAMPLES* section.

6.0 PICBasic Code Listing

Program compiles to 334 words with PBP 2.46.

```
'       DICE
'       ====
'       by Melanie (melanie@tekpumps.com)
'       12 March 2006

'       As if the world needed another Electronic Dice!
'       But it's something to do with a PIC10F chip...
'
'       For PIC10F202 or PIC10F206
'       Use Schematic & PCB Layout 9927-1

'       This circuit drives 7-LED's in classic Dice 'H' Configuration
'       Button ON with Dice Roll (and/or use Tilt Switch for "Shake 'n' Roll")
'       Auto Power OFF with standby current as low as 5uA
'       Will happilly run from a 3v Coin Cell or from up to 5.5vDC Power Source.
'       And all this from three and a half I/O's!

'
'       Configuration Fuse Definitions
'       ------------------------------
'       If compiling for a PIC16F202 change these DEVICE statements
'       and comment-out the CMCON0 line in the Initialise Section below.
'
@ DEVICE pic10F206, WDT_OFF
        ' Watchdog Timer Disabled
@ DEVICE pic10F206, MCLR_OFF
        ' Master Clear Options Internal
@ DEVICE pic10F206, PROTECT_OFF
        ' Code Protection Disabled

'
'       Hardware Defines
'       ----------------
LedX var GPIO.0
LedY var GPIO.1
LedZ var GPIO.2
Spin var GPIO.3

'
'       Software Defines
'       ----------------
CounterA var BYTE             ' Just a Counter
RollCount var WORD            ' Roll Counter

'
'       Software Constants
'       ------------------
DisplayTime con 150           ' Dice Display Duration in steps of 30mS

'
```

```
        '       Main Program Starts Here
        '       ========================

        '
        '       Initialise
        '       ----------
        CMCON0=%00000001              ' Comparator Disabled for PIC10F206 only
                                      ' Comment-Out the above line if compiling for PIC10F202
        TRISIO=%11111111              ' All pins Input
        STATUS=%10010000              ' Status Register - Reset on Wake on Pin Change
        OPTION_REG=%00000000          ' Enable GPWU and GPPU

        '
        '       Roll the Dice
        '       -------------
Start:
        While Spin=0                  ' Roll the Dice...
                RollCount=RollCount+1
                Wend
        For CounterA=0 to 200         ' Just for an extra Second or so...
                Pause 5               ' continue to monitor the Button...
                If Spin=0 then goto Start
                                      ' finger on button keeps on rolling...
                Next CounterA

        '
        '       Calculate Result
        '       ----------------
        Gosub LedsOff
        RollCount=RollCount//6

        '
        '       Display Result
        '       --------------
        For CounterA=0 to DisplayTime
                If RollCount=0 then   ' Number ONE Display
                        Gosub LedAOn
                        Gosub LedAOn
                        Pause 10
                        endif
                If RollCount=1 then   ' Number TWO Display
                        Gosub LedBOn
                        Gosub LedBon
                        Pause 10
                        endif
                If RollCount=2 then   ' Number THREE Display
                        Gosub LedAOn
                        Gosub LedBOn
                        Pause 10
                        endif
                If RollCount=3 then   ' Number FOUR Display
                        Gosub LedBOn
                        Gosub LedDOn
                        Pause 10
                        endif
                If RollCount=4 then   ' Number FIVE Display
                        Gosub LedAOn
                        Gosub LedBOn
                        Gosub LedDOn
                        endif
                If RollCount=5 then   ' Number SIX Display
                        Gosub LedBOn
                        Gosub LedCOn
                        Gosub LedDOn
                        endif
                Next CounterA
        If Spin=0 then goto Start     ' New Roll immediately follwing Result display

        '
        '       Quiesce until Button Pressed or shook/kicked (version with tilt switch)
        '       ------------------------------------------------------------------------
        CounterA=GPIO                 ' Read Input to Reset any pending Wake on Pin
        @ SLEEP                       ' Suspend and Quiesce
        Goto Start                    ' Life's a beach...

        '
        '       Subroutine Area
        '       ===============
```

```
        '
        '       Subroutine Enables LED-A (Centre One)
        '       -------------------------------------
LedAOn:
        High LedZ
        Low LedY
LedsOff:
        Pause 10
        TRISIO=%11111111
        Return

        '
        '       Subroutine Enables LED-B (Two of Two)
        '       -------------------------------------
LedBOn:
        Low LedZ
        High LedY
        Goto LedsOff

        '
        '       Subroutine Enables LED-C (Two of Six)
        '       -------------------------------------
LedCOn:
        High LedX
        Low LedY
        Goto LedsOff

        '
        '       Subroutine Enables LED-D (Two of Four)
        '       --------------------------------------
LedDOn:
        Low LedX
        High LedY
        Goto LedsOff

        End
```