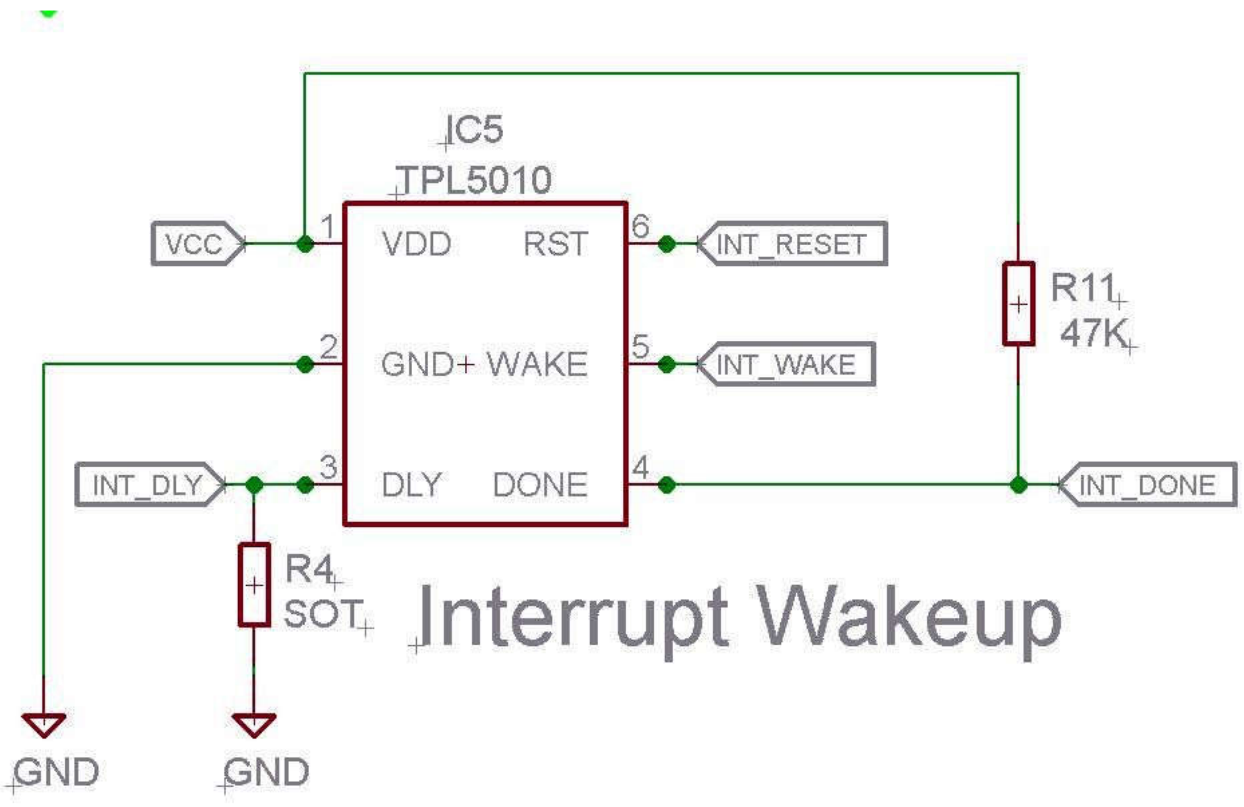
TPL5010 – Watchdog and Interrupt Timer

This is a description of some practical tests carries out on the operation of the TPL5010 low current timer. The timer uses a resistor on the DLY pin to set the timer, this period can be from a milli seconds up to 2 hours. See the appendix at the end of the document for resistor values and times.

The TPL5010 is similar to the TPL5110. The TPL5010 has the advantage of an open drain output on the RST pin which is ideal for direct drive of the RESET pin of a microcontroller.

The TPL5110 is designed to power down a project by driving a MOSFET or similar to connect and disconnect power to the microcontroller. This might seem a good choice for a project; deep sleep current in these circumstances should be in the 0.1uA region. However bear in mind that in these circumstances the project program starts from new every wakeup and thus the processor can be running for a couple of seconds before its ready to carry out its function. There is also a need for non-volatile storage to save status between sleeps. Deep sleeping the project so that the program continues just after the point where its put to sleep (with memory and variables intact) can in some circumstances be more power\battery efficient.



In the schematic above INT_DLY is normally connected to an input, that is swapped to an output briefly to reset the timer.

INT_RESET is connected to an input with internal pull-up enabled since this output from the TPL5010 is an open drain output.

INT_DONE is an input.

INT_WAKE is an output.

R4 sets the timer period, the value is read by the TPL5010 when power is applied to the device.

Note: R11 is a pull-up that ensures that in the event of the microcontroller failing to start, the DONE input of the TPL5010 does not float. If left floating it could be seen as a pulsing signal by the TPL5010 preventing the INT_RESET from operating.

A program was written that setup INT_WAKE and INT_RESET as interrupts, with flag set if the interrupts occurred. There were routines to pulse the INT_DONE and INT_DLY pins.

The program ran a loop that approximately every second printed elapsed seconds and checked the flags for the interrupts having occurred, and printed a count of these interrupts.

R4 was initially 12K giving a time period of around 12 seconds.

Test 1 – Effect of no INT_DONE pulses

R4 was 12K giving a time period of around 12 seconds. Apply power, no pulses at start-up to INT_DLY or INT_DONE.

After 12 seconds there is an INT_RESET. INT_RESET interrupts then occur every 12 seconds, there are no INT_WAKE interrupts.

Test 2 - Effect of pulsing INT_DONE at power up.

R4 was 12K giving a time period of around 12 seconds. Apply power, pulse INT_DONE at start-up.

After 12 seconds there is an INT_WAKE interrupt then 12 seconds later an INT_RESET interrupt. INT_RESETs then occur every 12 seconds, there are no further INT_WAKE interrupts.

Test 3 – Effect of regular INT_DONE pulses.

R4 was 12K giving a time period of around 12 seconds. Apply power, pulse INT_DONE at start-up. After the INT_WAKE interrupt pulse INT_DONE .

After 12 seconds there is an INT_WAKE interrupt then every 12 seconds there is a further INT_WAKE interrupt. There are no INT_RESET interrupts. .

Test 4 – Effect of DONE pin floating.

R4 was 12K giving a time period of around 12 seconds. Apply power, leave INT_DONE floating.

There are erratic INT_WAKE interrupts, less than 12 seconds apart. There are no INT_RESET interrupts.

Therefore if the microcontroller fails to start correctly for some reason and INT_WAKE is not changed from the default of input to an output there will be no reset output which could recover the situation. There does not seem to be a reason as to why the WAKE pin on the TPL5010 does not have a weak pull-up.

Test 5 – Pulse DLY timer pin with IO pin.

R4 was 47K giving a time period of around 360 seconds. Apply power, pulse INT_DONE at start-up. After the first INT_WAKE or INT_RESET interrupt, pulse INT_DLY.

The pulse on INT_DLY resets the counters in the TPL5010 and causes the resistor to be read and checked again after the high on the INT_DLY pin is removed. As this is a valid reset operation (assuming INT_DLY is high for more than 20mS) there is also low pulse on the TPL5010 RST output (INT_RESET).

So the sequence is; an INT_WAKE interrupt which causes an INT_RESET interrupt and there is then a sequence of INT_RESET interrupts every 370 seconds with no further INT_WAKE interrupts.

Test 6 – Effect of IO pin on DLY timer pin.

R4 was 47K giving a time period of around 360 seconds with INT_RESET connected to the processor RESET line. Check the time period a) with an IO line connected to DLY and b) without an IO line connected to DLY.

a) Apply power, pulse INT_DONE once at start-up. After power up there is an INT_WAKE at 367 seconds and a reset of the microcontroller after 737 seconds. The sequence of INT_WAKE interrupts and resets then continues.

b) Repeat a) with an IO line as input connected to INT_DLY. Same sequence as for a) at 369seconds and.
Conclusion; there is no particular problem with connecting an IO line to DLY to force a reset\re-read of the timer resistor, but its not necessary. A simple bit of code can be used to reset the microcontroller.

Test 7 – Total sleep current.

Atmega 1284P using TP5010 as interrupt wake up timer and MCP1700 3V3 regulator. R4 was 12K giving a time period of around 12 seconds, 1284P put into deep sleep to wakeup on TPL5010 WAKE output.
Using following libraries;

```
#include "LowPower.h"              //https://github.com/rocketscream/Low-Power
#include "PinChangeInterrupt.h"  //https://github.com/NicoHood/PinChangeInterrupt
```

Atmega1284 wakes up and goes back to sleep every 12 seconds, deep sleep current 2uA.

Stuart Robinson

September 2018

Appendix A

Extracts from TPL5010 datasheet;

http://www.ti.com/lit/ds/symlink/tpl5010.pdf

Table 2. First 9 Time Intervals

tIP (ms)	Resistance (Ω)	Closest Real Value (Ω)	Parallel of Two 1% Tolerance Resistors, (kΩ)
100	500	500	1.0 // 1.0
200	1000	1000	-
300	1500	1500	2.43 // 3.92
400	2000	2000	-
500	2500	2500	4.42 // 5.76
600	3000	3000	5.36 // 6.81
700	3500	3500	4.75 // 13.5
800	4000	4000	6.19 // 11.3
900	4500	4501	6.19 // 16.5

Table 3. Most Common Time Intervals Between 1s to 2h

tIP	Calculated Resistance (kΩ)	Closest Real Value (kΩ)	Parallel of Two 1% Tolerance Resistors, (kΩ)
1s	5.20	5.202	7.15 // 19.1
2s	6.79	6.788	12.4 // 15.0
3s	7.64	7.628	12.7// 19.1
4s	8.30	8.306	14.7 // 19.1
5s	8.85	8.852	16.5 // 19.1
6s	9.27	9.223	18.2 // 18.7
7s	9.71	9.673	19.1 // 19.6
8s	10.18	10.180	11.5 // 8.87
9s	10.68	10.68	17.8 // 26.7
10s	11.20	11.199	15.0 // 44.2
20s	14.41	14.405	16.9 // 97.6
30s	16.78	16.778	32.4 // 34.8
40s	18.75	18.748	22.6 // 110.0
50s	20.047	20.047	28.7 // 66.5

Table 3. Most Common Time Intervals Between 1s to 2h (continued)

tIP	Calculated Resistance (kΩ)	Closest Real Value (kΩ)	Parallel of Two 1% Tolerance Resistors, (kΩ)
1min	22.02	22.021	40.2 // 48.7
2min	29.35	29.349	35.7 // 165.0
3min	34.73	34.729	63.4 // 76.8

4min	39.11	39.097	63.4 // 102.0
5min	42.90	42.887	54.9 // 196.0
6min	46.29	46.301	75.0 // 121.0
7min	49.38	49.392	97.6 // 100.0
8min	52.24	52.224	88.7 // 127.0
9min	54.92	54.902	86.6 // 150.0
10min	57.44	57.437	107.0 // 124.0
20min	77.57	77.579	140.0 // 174.0
30min	92.43	92.233	182.0 // 187.0
40min	104.67	104.625	130.0 // 536.00
50min	115.33	115.331	150.0 // 499.00
1h	124.91	124.856	221.0 // 287.00
1h30min	149.39	149.398	165.0 // 1580.0
2h	170.00	170.00	340.0 // 340.0

The End.