

# CPE 325: Embedded Systems Laboratory

## Laboratory #3 Tutorial

### Digital I/O on Experimenter's Board: LEDs and Switches

Aleksandar Milenković

Email: [milenka@uah.edu](mailto:milenka@uah.edu)

Web: <http://www.ece.uah.edu/~milenka>

#### Objective:

This tutorial will help introduce MSP430 parallel ports and how they are used for interfacing LEDs and switches. Specifically, you will learn the following topics:

*Hardware development platform MSP-EXP430F5539LP Experimenter Board*  
*I/O Interfacing Using Parallel Ports (LEDs and switches)*  
*Software delays (time estimation)*

#### Notes:

All previous tutorials are required for successful completion of this lab, especially, the tutorials introducing the TI Experimenter's board and the Code Composer Studio software development environment.

**Recommended reading:** Introduction to the TI's MSP-EXP430F5529 LaunchPad Experimenter's Board User Guide <http://www.ti.com/lit/ug/slau533d/slau533d.pdf>.

#### Contents:

1	Digital Input/Output Introduction .....	2
2	Turning on a LED Project Using C Language .....	3
3	Blinking LEDs Using C Language .....	5
4	Interfacing Buttons (Switches) .....	7
5	References .....	9

# 1 Digital Input/Output Introduction

A microcontroller interacts in many ways with the system in which it is embedded. It may receive inputs from a human through switches, buttons, touch sensors, keypads, or microphone. In the opposite direction, the microcontroller may control external devices such as light-emitting diodes (LEDs), seven-segment displays, liquid-crystal displays (LCDs), or actuators (e.g., motors). The MSP430 microcontrollers can drive these external devices directly if they work at the same voltage and draw a sufficiently small current that microcontroller is able to provide. Heavier loads require dedicated circuitry to drive them.

The most straightforward form of input/output is through the digital input/output ports using binary values (0 or 1). The primary way how the MSP430 interfaces the rest of the world is through 8-bit parallel ports. The actual number of physical parallel ports varies with device type and can range from two to ten 8-bit parallel ports, typically marked as Px, x=1...10. The MSP430 parallel ports are bit-configurable (every bit can be configured as an input or output) and can work as standard digital input/outputs or as special-function ports (e.g., serve as analog inputs for an analog-to-digital converter or timer output). The parallel ports are directly connected to the chip pins. A parallel port encompasses multiple registers, including:

- PxIN - input register, reading it returns the logical values on the pins (determined by the external signals). These registers are read-only and bit value of 0 indicates that the corresponding input is low and bit value of 1 indicates that the input is high.
- PxOUT - output register, writing it sends the value to the corresponding port pin when the pin is configured for the I/O function with output direction. Bit value of 0 will produce low output voltage and bit value of 1 will produce high output voltage.
- PxDIR - direction register, configures the direction of the corresponding I/O pins (e.g., P2DIR=0xFC = 11111100b configures bits 1 and 0 of port P2 as input pins and all other pins are outputs).
- PxSEL - selection register, setting bits in this register allows the user to change the port pin function from the standard digital I/O to its corresponding special function. MSP430 interfaces external world predominantly through parallel ports and their default operation is a standard digital input/output (PxSEL=0x00). However, some of the pins have an alternative special function – e.g., they can act as an analog input channel (A0) to the analog-to-digital converter or serial data output of a serial communication interface (TDO). The reference manual specifies special functions for each port pin. They are highly device-specific – developers have to consult the reference manual for the microcontroller they are using.
- PxREN – enables the pull-up or pull-down resistor configuration (e.g, P2REN = 0x2 enables the pull-up resistor on P2.1 that is connected in a series with switch SW1 on the MSP-EXP430F5529LP board.)
- Ports P1 and P2 also have ability to serve as sources of interrupts and several registers are associated with this function. These are:
  - PxIE – Port x Interrupt Enable register for enabling/disabling interrupts (x=1, 2),
  - PxIFG – Port x Interrupt Flag register for tracking pending requests,

- PxIES – Port x Interrupt Edge Select register for selecting type of event that triggers an interrupt – rising edge at the port input (0 -> 1) or falling edge (1 -> 0);
- PxIV – Port x Interrupt Vector Word. All interrupts associated with a single port share a single interrupt service routine. The highest priority enabled pending interrupt request generates a number in the PxIV register. This number can be used by the code in the corresponding interrupt service routine to speed up interrupt processing.

Our development platform includes two LEDs (LED1-LED2) and two switches (SW1 and SW2). The LEDs can be turned on and off by writing digital 1 or 0 to the appropriate output port registers, respectively. Therefore, in order to turn a LED on, first the I/O port should be set to output direction, and then either a 0 or a 1 should be written to the output register. In this lab, LEDs are turned on and off with specific frequencies. Let us develop a program to blink a LED.

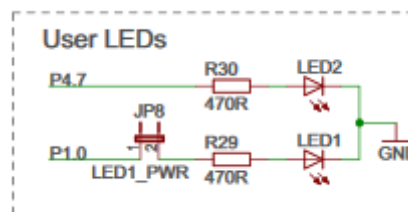
## 2 Turning on a LED Project Using C Programming Language

This section defines the problem that will be solved by the "Turn on a LED" application. Your task is to write a C program that will turn on the LED1 on the TI's EXP-MSP430F5529LP Experimenter Board.

Step 1. Analyze the assignment.

In order to better understand the problem, we will first study schematics of the MSP430F5529LP Experimenter Board. This board includes TI's MSP430 microcontroller (MSP430F5529), 2 leds (LED1-LED2), 2 switches (SW1 and SW2), and several extension slots that allow an easy access to all microcontroller ports. A detailed schematic of the board is provided in the following document: <http://www.ti.com/lit/ug/slau533d/slau533d.pdf>. Go to page 55 and locate images with headings "User Buttons" and "User LEDs". In the schematic see how the ports are actually connected to physical LEDs (a diode through a resistor Figure 1).

*What microcontroller port pins are connected to LED1 and LED2? A LED is ON when the current is flowing through it and it is OFF when there is no current. How should we drive the corresponding ports to have the current flow?*



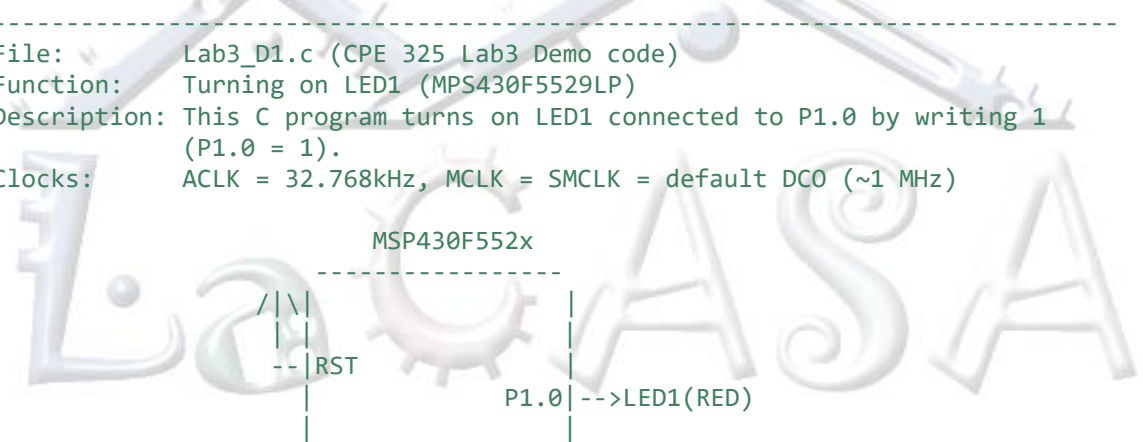
**Figure 1. LED1 and LED2 connections on the TI's experimenter's board.**

Step 2. Develop a plan.

From the schematic we see that if we want LED1 on, we should provide a logical '1' at the output port of the microcontroller (port P1.0), and a logical '0' if we want LED1 to be off. We

could take several approaches to solving this problem. Figure 2 illustrates one such approach - after initializing the port P1.0 as output (P1DIR=00000001), setting P1.0 to logic '1', the program will spend all its time in an infinite loop (Figure 2).

Though rather a simple program, it demonstrates main aspects of a program targeting embedded platforms. It starts with the instructions to initialize and configure the peripherals. In our case that means to stop the watchdog timer (line 24), configures port 1, bit 0 to output direction (line 25), and set the P1OUT to 0x01, providing high voltage on P1.0 pin and thus current to flow through the LED1. Please note that the watchdog timer in MSP430 is enabled by default and will cause reset every ~33 ms if left active interfering with your program execution. So please remember, always turn off the watchdog timer if you are not using using the statement from line 24. The main program in our case has nothing left to do, so it is an infinite loop in line 27. An alternative to the forever loop is to put the microcontroller into a low-power mode that suspends program execution, saving energy consumed by your platform while providing the steady current to LED1.



```

1  /*-----
2  * File:      Lab3_D1.c (CPE 325 Lab3 Demo code)
3  * Function:   Turning on LED1 (MPS430F5529LP)
4  * Description: This C program turns on LED1 connected to P1.0 by writing 1
5  *              (P1.0 = 1).
6  * Clocks:    ACLK = 32.768kHz, MCLK = SMCLK = default DCO (~1 MHz)
7  *
8  *              MSP430F552x
9  *
10 *              /\|
11 *              | |
12 *              --| RST
13 *
14 *              |
15 *              |
16 *              |
17 *              |
18 *              |
19 *              |
20 *              |
21 *              |
22 *              |
23 *              |
24 *              |
25 *              |
26 *              |
27 *              |
28 *              |
29 *              |
30 *              |
31 *              |
32 *              |
33 *              |
34 *              |
35 *              |
36 *              |
37 *              |
38 *              |
39 *              |
40 *              |
41 *              |
42 *              |
43 *              |
44 *              |
45 *              |
46 *              |
47 *              |
48 *              |
49 *              |
50 *              |
51 *              |
52 *              |
53 *              |
54 *              |
55 *              |
56 *              |
57 *              |
58 *              |
59 *              |
60 *              |
61 *              |
62 *              |
63 *              |
64 *              |
65 *              |
66 *              |
67 *              |
68 *              |
69 *              |
70 *              |
71 *              |
72 *              |
73 *              |
74 *              |
75 *              |
76 *              |
77 *              |
78 *              |
79 *              |
80 *              |
81 *              |
82 *              |
83 *              |
84 *              |
85 *              |
86 *              |
87 *              |
88 *              |
89 *              |
90 *              |
91 *              |
92 *              |
93 *              |
94 *              |
95 *              |
96 *              |
97 *              |
98 *              |
99 *              |
100 *              |
101 *              |
102 *              |
103 *              |
104 *              |
105 *              |
106 *              |
107 *              |
108 *              |
109 *              |
110 *              |
111 *              |
112 *              |
113 *              |
114 *              |
115 *              |
116 *              |
117 *              |
118 *              |
119 *              |
120 *              |
121 *              |
122 *              |
123 *              |
124 *              |
125 *              |
126 *              |
127 *              |
128 *              |
129 *              |
130 *              |
131 *              |
132 *              |
133 *              |
134 *              |
135 *              |
136 *              |
137 *              |
138 *              |
139 *              |
140 *              |
141 *              |
142 *              |
143 *              |
144 *              |
145 *              |
146 *              |
147 *              |
148 *              |
149 *              |
150 *              |
151 *              |
152 *              |
153 *              |
154 *              |
155 *              |
156 *              |
157 *              |
158 *              |
159 *              |
160 *              |
161 *              |
162 *              |
163 *              |
164 *              |
165 *              |
166 *              |
167 *              |
168 *              |
169 *              |
170 *              |
171 *              |
172 *              |
173 *              |
174 *              |
175 *              |
176 *              |
177 *              |
178 *              |
179 *              |
180 *              |
181 *              |
182 *              |
183 *              |
184 *              |
185 *              |
186 *              |
187 *              |
188 *              |
189 *              |
190 *              |
191 *              |
192 *              |
193 *              |
194 *              |
195 *              |
196 *              |
197 *              |
198 *              |
199 *              |
200 *              |
201 *              |
202 *              |
203 *              |
204 *              |
205 *              |
206 *              |
207 *              |
208 *              |
209 *              |
210 *              |
211 *              |
212 *              |
213 *              |
214 *              |
215 *              |
216 *              |
217 *              |
218 *              |
219 *              |
220 *              |
221 *              |
222 *              |
223 *              |
224 *              |
225 *              |
226 *              |
227 *              |
228 *              |
229 *              |
230 *              |
231 *              |
232 *              |
233 *              |
234 *              |
235 *              |
236 *              |
237 *              |
238 *              |
239 *              |
240 *              |
241 *              |
242 *              |
243 *              |
244 *              |
245 *              |
246 *              |
247 *              |
248 *              |
249 *              |
250 *              |
251 *              |
252 *              |
253 *              |
254 *              |
255 *              |
256 *              |
257 *              |
258 *              |
259 *              |
260 *              |
261 *              |
262 *              |
263 *              |
264 *              |
265 *              |
266 *              |
267 *              |
268 *              |
269 *              |
270 *              |
271 *              |
272 *              |
273 *              |
274 *              |
275 *              |
276 *              |
277 *              |
278 *              |
279 *              |
280 *              |
281 *              |
282 *              |
283 *              |
284 *              |
285 *              |
286 *              |
287 *              |
288 *              |
289 *              |
290 *              |
291 *              |
292 *              |
293 *              |
294 *              |
295 *              |
296 *              |
297 *              |
298 *              |
299 *              |
300 *              |
301 *              |
302 *              |
303 *              |
304 *              |
305 *              |
306 *              |
307 *              |
308 *              |
309 *              |
310 *              |
311 *              |
312 *              |
313 *              |
314 *              |
315 *              |
316 *              |
317 *              |
318 *              |
319 *              |
320 *              |
321 *              |
322 *              |
323 *              |
324 *              |
325 *              |
326 *              |
327 *              |
328 *              |
329 *              |
330 *              |
331 *              |
332 *              |
333 *              |
334 *              |
335 *              |
336 *              |
337 *              |
338 *              |
339 *              |
340 *              |
341 *              |
342 *              |
343 *              |
344 *              |
345 *              |
346 *              |
347 *              |
348 *              |
349 *              |
350 *              |
351 *              |
352 *              |
353 *              |
354 *              |
355 *              |
356 *              |
357 *              |
358 *              |
359 *              |
360 *              |
361 *              |
362 *              |
363 *              |
364 *              |
365 *              |
366 *              |
367 *              |
368 *              |
369 *              |
370 *              |
371 *              |
372 *              |
373 *              |
374 *              |
375 *              |
376 *              |
377 *              |
378 *              |
379 *              |
380 *              |
381 *              |
382 *              |
383 *              |
384 *              |
385 *              |
386 *              |
387 *              |
388 *              |
389 *              |
390 *              |
391 *              |
392 *              |
393 *              |
394 *              |
395 *              |
396 *              |
397 *              |
398 *              |
399 *              |
400 *              |
401 *              |
402 *              |
403 *              |
404 *              |
405 *              |
406 *              |
407 *              |
408 *              |
409 *              |
410 *              |
411 *              |
412 *              |
413 *              |
414 *              |
415 *              |
416 *              |
417 *              |
418 *              |
419 *              |
420 *              |
421 *              |
422 *              |
423 *              |
424 *              |
425 *              |
426 *              |
427 *              |
428 *              |
429 *              |
430 *              |
431 *              |
432 *              |
433 *              |
434 *              |
435 *              |
436 *              |
437 *              |
438 *              |
439 *              |
440 *              |
441 *              |
442 *              |
443 *              |
444 *              |
445 *              |
446 *              |
447 *              |
448 *              |
449 *              |
450 *              |
451 *              |
452 *              |
453 *              |
454 *              |
455 *              |
456 *              |
457 *              |
458 *              |
459 *              |
460 *              |
461 *              |
462 *              |
463 *              |
464 *              |
465 *              |
466 *              |
467 *              |
468 *              |
469 *              |
470 *              |
471 *              |
472 *              |
473 *              |
474 *              |
475 *              |
476 *              |
477 *              |
478 *              |
479 *              |
480 *              |
481 *              |
482 *              |
483 *              |
484 *              |
485 *              |
486 *              |
487 *              |
488 *              |
489 *              |
490 *              |
491 *              |
492 *              |
493 *              |
494 *              |
495 *              |
496 *              |
497 *              |
498 *              |
499 *              |
500 *              |
501 *              |
502 *              |
503 *              |
504 *              |
505 *              |
506 *              |
507 *              |
508 *              |
509 *              |
510 *              |
511 *              |
512 *              |
513 *              |
514 *              |
515 *              |
516 *              |
517 *              |
518 *              |
519 *              |
520 *              |
521 *              |
522 *              |
523 *              |
524 *              |
525 *              |
526 *              |
527 *              |
528 *              |
529 *              |
530 *              |
531 *              |
532 *              |
533 *              |
534 *              |
535 *              |
536 *              |
537 *              |
538 *              |
539 *              |
540 *              |
541 *              |
542 *              |
543 *              |
544 *              |
545 *              |
546 *              |
547 *              |
548 *              |
549 *              |
550 *              |
551 *              |
552 *              |
553 *              |
554 *              |
555 *              |
556 *              |
557 *              |
558 *              |
559 *              |
560 *              |
561 *              |
562 *              |
563 *              |
564 *              |
565 *              |
566 *              |
567 *              |
568 *              |
569 *              |
570 *              |
571 *              |
572 *              |
573 *              |
574 *              |
575 *              |
576 *              |
577 *              |
578 *              |
579 *              |
580 *              |
581 *              |
582 *              |
583 *              |
584 *              |
585 *              |
586 *              |
587 *              |
588 *              |
589 *              |
590 *              |
591 *              |
592 *              |
593 *              |
594 *              |
595 *              |
596 *              |
597 *              |
598 *              |
599 *              |
600 *              |
601 *              |
602 *              |
603 *              |
604 *              |
605 *              |
606 *              |
607 *              |
608 *              |
609 *              |
610 *              |
611 *              |
612 *              |
613 *              |
614 *              |
615 *              |
616 *              |
617 *              |
618 *              |
619 *              |
620 *              |
621 *              |
622 *              |
623 *              |
624 *              |
625 *              |
626 *              |
627 *              |
628 *              |
629 *              |
630 *              |
631 *              |
632 *              |
633 *              |
634 *              |
635 *              |
636 *              |
637 *              |
638 *              |
639 *              |
640 *              |
641 *              |
642 *              |
643 *              |
644 *              |
645 *              |
646 *              |
647 *              |
648 *              |
649 *              |
650 *              |
651 *              |
652 *              |
653 *              |
654 *              |
655 *              |
656 *              |
657 *              |
658 *              |
659 *              |
660 *              |
661 *              |
662 *              |
663 *              |
664 *              |
665 *              |
666 *              |
667 *              |
668 *              |
669 *              |
670 *              |
671 *              |
672 *              |
673 *              |
674 *              |
675 *              |
676 *              |
677 *              |
678 *              |
679 *              |
680 *              |
681 *              |
682 *              |
683 *              |
684 *              |
685 *              |
686 *              |
687 *              |
688 *              |
689 *              |
690 *              |
691 *              |
692 *              |
693 *              |
694 *              |
695 *              |
696 *              |
697 *              |
698 *              |
699 *              |
700 *              |
701 *              |
702 *              |
703 *              |
704 *              |
705 *              |
706 *              |
707 *              |
708 *              |
709 *              |
710 *              |
711 *              |
712 *              |
713 *              |
714 *              |
715 *              |
716 *              |
717 *              |
718 *              |
719 *              |
720 *              |
721 *              |
722 *              |
723 *              |
724 *              |
725 *              |
726 *              |
727 *              |
728 *              |
729 *              |
730 *              |
731 *              |
732 *              |
733 *              |
734 *              |
735 *              |
736 *              |
737 *              |
738 *              |
739 *              |
740 *              |
741 *              |
742 *              |
743 *              |
744 *              |
745 *              |
746 *              |
747 *              |
748 *              |
749 *              |
750 *              |
751 *              |
752 *              |
753 *              |
754 *              |
755 *              |
756 *              |
757 *              |
758 *              |
759 *              |
760 *              |
761 *              |
762 *              |
763 *              |
764 *              |
765 *              |
766 *              |
767 *              |
768 *              |
769 *              |
770 *              |
771 *              |
772 *              |
773 *              |
774 *              |
775 *              |
776 *              |
777 *              |
778 *              |
779 *              |
780 *              |
781 *              |
782 *              |
783 *              |
784 *              |
785 *              |
786 *              |
787 *              |
788 *              |
789 *              |
790 *              |
791 *              |
792 *              |
793 *              |
794 *              |
795 *              |
796 *              |
797 *              |
798 *              |
799 *              |
800 *              |
801 *              |
802 *              |
803 *              |
804 *              |
805 *              |
806 *              |
807 *              |
808 *              |
809 *              |
810 *              |
811 *              |
812 *              |
813 *              |
814 *              |
815 *              |
816 *              |
817 *              |
818 *              |
819 *              |
820 *              |
821 *              |
822 *              |
823 *              |
824 *              |
825 *              |
826 *              |
827 *              |
828 *              |
829 *              |
830 *              |
831 *              |
832 *              |
833 *              |
834 *              |
835 *              |
836 *              |
837 *              |
838 *              |
839 *              |
840 *              |
841 *              |
842 *              |
843 *              |
844 *              |
845 *              |
846 *              |
847 *              |
848 *              |
849 *              |
850 *              |
851 *              |
852 *              |
853 *              |
854 *              |
855 *              |
856 *              |
857 *              |
858 *              |
859 *              |
860 *              |
861 *              |
862 *              |
863 *              |
864 *              |
865 *              |
866 *              |
867 *              |
868 *              |
869 *              |
870 *              |
871 *              |
872 *              |
873 *              |
874 *              |
875 *              |
876 *              |
877 *              |
878 *              |
879 *              |
880 *              |
881 *              |
882 *              |
883 *              |
884 *              |
885 *              |
886 *              |
887 *              |
888 *              |
889 *              |
890 *              |
891 *              |
892 *              |
893 *              |
894 *              |
895 *              |
896 *              |
897 *              |
898 *              |
899 *              |
900 *              |
901 *              |
902 *              |
903 *              |
904 *              |
905 *              |
906 *              |
907 *              |
908 *              |
909 *              |
910 *              |
911 *              |
912 *              |
913 *              |
914 *              |
915 *              |
916 *              |
917 *              |
918 *              |
919 *              |
920 *              |
921 *              |
922 *              |
923 *              |
924 *              |
925 *              |
926 *              |
927 *              |
928 *              |
929 *              |
930 *              |
931 *              |
932 *              |
933 *              |
934 *              |
935 *              |
936 *              |
937 *              |
938 *              |
939 *              |
940 *              |
941 *              |
942 *              |
943 *              |
944 *              |
945 *              |
946 *              |
947 *              |
948 *              |
949 *              |
950 *              |
951 *              |
952 *              |
953 *              |
954 *              |
955 *              |
956 *              |
957 *              |
958 *              |
959 *              |
960 *              |
961 *              |
962 *              |
963 *              |
964 *              |
965 *              |
966 *              |
967 *              |
968 *              |
969 *              |
970 *              |
971 *              |
972 *              |
973 *              |
974 *              |
975 *              |
976 *              |
977 *              |
978 *              |
979 *              |
980 *              |
981 *              |
982 *              |
983 *              |
984 *              |
985 *              |
986 *              |
987 *              |
988 *              |
989 *              |
990 *              |
991 *              |
992 *              |
993 *              |
994 *              |
995 *              |
996 *              |
997 *              |
998 *              |
999 *              |
1000 *             */
1001 #include <msp430.h>
1002
1003 void main(void)
1004 {
1005     WDTCTL = WDTPW + WDTHOLD;    // Stop watchdog timer
1006     P1DIR |= 0x01;               // Set P1.0 to output direction (0000_0001)
1007     P1OUT |= 0x01;               // Set P1OUT to 0000_0001b (LED1 is ON)
1008     for (;;)                    // Infinite loop so that the LED stays ON forever
1009     {
1010     }
1011 }

```

**Figure 2. Turn-on an LED Using C Code (Lab3\_D1.c)**

### 3 Blinking LEDs Using C Programming Language

This section walks you through the "Blink the LEDs" application. Your task is to write a C program that will alternately blink the LED1 and LED2 on the TI's EXP-MSP430F5529LP board with 1 Hz frequency, i.e., the LED1/LED2 will be on/off for about 0.5 s.

We could take several approaches to blink the LEDs. The simplest one is to toggle the port P1.0 and P4.7 and have 0.5 seconds delay in software as shown in Figure 3 (Lab3\_D2.c). After initializing the microcontroller, our program will spend all its time in an infinite loop (LED1 and LED2 should be repeatedly blinking alternately). Inside a loop we will toggle the ports P1.0 and P4.7 and then wait for approximately 0.5s. The port toggling can be done using an XOR operation of the current value of the port (P1OUT) and the constant 0x01, i.e., (P1OUT=P1OUT xor 0x01). Software delay of 0.5s can be implemented using an empty loop with a certain number of iterations (see the for loop in Figure 3).

To exactly calculate the software delay we need to know the number of clock cycles to execute one iteration of the for loop and the clock cycle time. The total number of clock cycles taken to execute the entire loop can be calculated by multiplying the number of clock cycles taken for one iteration of the loop with the number of iterations. In Figure 3, one iteration of the for loop takes 10 clock cycles and the loop counter is 50,000. Thus, it takes  $10 \times 50,000 = 500,000$  clock cycles to execute the for loop (Note: when the counter is initialized in the for loop, first iteration takes some extra clock cycles. Since this is only for one time, it is not considered in our calculation for simplicity). Make sure you have your optimization turned-off. Sometimes the delay may differ because of optimizations and you may not observe the desired delay.

Determining clock cycle time requires in-depth understanding of the Unified Clock System (UCS) module of the MSP430 which is beyond the scope of this tutorial. We note that the processor clock frequency is approximately 1 MHz (1,048,576 Hz to be exact) for our configuration, so the clock cycle time is  $1\mu\text{s}$ . The total delay is thus  $500,000 \times 1\mu\text{s} = 0.5\text{s}$ .

```
1  /*-----  
2  * File:      Lab3_D2.c (CPE 325 Lab3 Demo code)  
3  * Function:   Blinking LED1 and LED2 (MPS430F5529)  
4  * Description: This C program toggle LED1 and LED2 at 1Hz by xoring P1.0 and  
5  *             P4.7 inside a loop. The LEDs are on when P1.0=1 and P4.7=1.  
6  *             The LED1 is initialized to be off and LED2 to be on.  
7  * Clocks:    ACLK = 32.768kHz, MCLK = SMCLK = default DCO (~1 MHz)  
8  *  
9  *  
10 *             MSP430F552x  
11 *             -----  
12 *             /|\|  
13 *             --| RST  
14 *             |  
15 *             | P1.0 --> LED1(RED)  
16 *             | P4.7 --> LED2(GREEN)  
17 * Input:      None  
18 * Output:     LED1 and LED2 blink alternately at 1 Hz frequency
```





```

15 * | P4.7| ---> Green LED
16 *
17 * Author: Aleksandar Milenkovic, milenkovic@computer.org
18 * Date: August 2, 2020
19 *
20 *****/
21 #include <msp430.h>
22
23 #define REDLED 0x01 // mask for BIT0=00000001b
24 #define GREENLED 0x80 // mask for BIT7=10000000b
25
26 void main(void) {
27
28     WDTCTL = WDTPW | WDTHOLD; // stop watchdog timer
29     P1DIR |= REDLED; // configure P1.0 as output
30     P4DIR |= GREENLED; // configure P4.7 as output
31
32     P1OUT = P1OUT & ~REDLED; // P1.0 is cleared
33     P4OUT = P4OUT | GREENLED; // P4.7 is on
34     while(1) { // main application loop
35         _delay_cycles(500000); // delay of ~500 ms
36         P1OUT ^= REDLED; // toggle P1.0
37         P4OUT ^= GREENLED; // toggle P4.7
38     }
39 }
40

```

Figure 4. Blinking the LEDs Every Second (BlinkTheLEDs.c)

## 4 Interfacing Buttons (Switches)

Often, we would like to trigger a certain task in embedded systems by pressing a button or switch. Here we will learn how to interface a switch, how to detect that it is pressed and how to detect that it is released. First, let us look at the MSP-EXP430F5529LP platform schematic (Figure 5) that illustrates how the buttons are connected to the MSP430. We can see two switches S1 and S2. One terminal is connected to the ground and the other to P2.1 for the S1 button and to P1.1 for the S2 button. (see Figure 5). When the buttons are not pressed the inputs P2.1 and P1.1 are not connected to the ground. The port pins are configured with the output pull-up resistor driving high voltage on P2.1 and P1.1 – thus the input will see logic ‘1’ when buttons are not pressed. When the buttons are pressed, P2.1 and P1.1 are connected to the ground – the current flows through the pull up resistor and input register reads logic ‘0.’

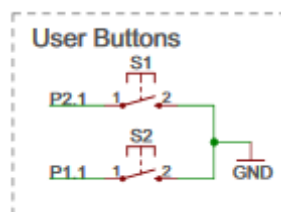
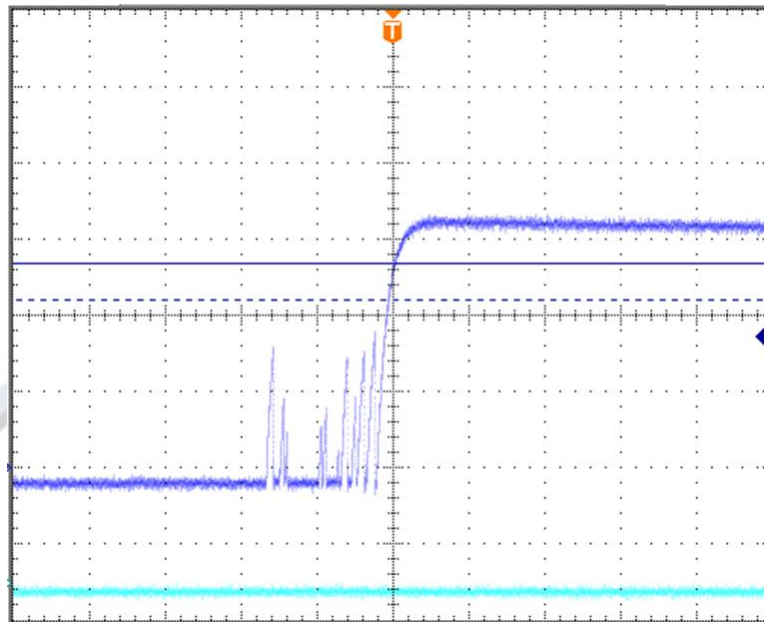


Figure 5. MSP-EXP430F5529LP Board Buttons Schematic

When interfacing switches we must take care that we properly detect whether a switch is pressed. Typically, we need (i) to detect that a switch has been pressed, (ii) to debounce it (apply software delay to ensure that it is indeed pressed, rather than a faulty detection caused by noise), and (iii) to detect that it has been released. An example of the noise created by the action of a switch pressing can be seen in Figure 6 (note that in this example, pressed switch generates logic 1). If not programmed correctly, our application could think that each spike was an individual press of the switch, which could be detrimental to the functionality of the application.



**Figure 6. Typical Oscilloscope Reading of Switch Input**

We can individually examine the P2IN bit 1 status to see if S1 has been pressed. If it has indeed been pressed (bit 1 of P2IN is 0), we execute a loop to wait for a short period of time (~20 ms) to avoid faulty detections that may be caused by electrical noise. After the software delay, we validate that the input port pin is still at 0; if yes, that means that the button S1 is indeed pressed.

We may keep the button pressed for a longer period of time, and we often want to ensure that a switch is released before we go to process an event that may be triggered by this detection. Figure 7 shows a program that turns LED1 on when the button S1 is pressed and it keeps it on as long as S1 is pressed. When the button S1 is released, LED1 is turned off.

```

1  /*-----
2  * File:      Lab3_D3.c (CPE 325 Lab3 Demo code)
3  * Function:   Turning on LED1 when S1 is pressed (MSP-EXP430F5529LP)
4  * Description: This C program turns on LED1 connected to P1.0 when the S1 is
5  *              pressed. S1 is connected to P2.1 and when it is pressed,
6  *              P2IN, bit 1 is read as a logic 0 (check the schematic).
7  *              To avoid faulty detection of button press,
8  *              debouncing delay of 20ms is added before turning on the LED1.

```





- Chapter 7 in the John H. Davies' MSP430 Microcontroller Basics

