# PROG1360 Assignment 5 – Watchdogs and LEDs

Individual or Pair

Please note: Copying and submitting work that is not your own is an academic violation, and will be dealt with as such.

## Objectives

* Interact with the LEDs directly using memory
* Implement and demonstrate the use of the watchdog
* Structure your code

## Major Tasks

### Demonstration – Building on Labs

This assignment builds on the labs and assignments so far.

The Demonstration section of this document outlines the items you are required to show.

## Tasks

You will revisit the blinking lights game of Assignment 3 and make modifications. Modifications to make:

1. Replace the calls to BSP\_LED\_Toggle / LED\_ON / LED\_OFF with direct memory access as shown in Lab 9.
2. Modify the winning code (blink all lights on and off twice) with direct memory access, changing the state of all lights with single calls (all on / all off) instead of individual calls.

Additions to make:

Create a new test item in the menu (the one we normally access with minicom) to demonstrate that the hardware watchdog works. Requirements:

1. Initialize and start the watchdog from the C code as shown in the lab
2. Experiment with the prescaler and reload values to determine the watchdog duration. Make this duration configurable in the call to your C code.
3. This function will blink the LEDs in a continuous loop using direct memory access
4. If the user button is not pressed, refresh the watchdog between LED blinks
5. If the user button is pressed, don’t refresh the watchdog, but continue blinking the LEDs. A properly configured watchdog will result in the board rebooting after the watchdog timer has expired.

Given the command below (where xx is your initials)

xxWatch duration delay

your board will execute the blinking light loop with the watchdog enabled for the duration you specify. You will need to determine what the units of duration are.

Where necessary, you can still use the busy\_delay. You may have to refresh the watchdog during the busy delay if the timeout is less than the delay.

### Requirements

* Only the watchdog initialization is to be implemented in your C hook code.
* You may use library functions from the stm32 code to parse input.
* You must use direct memory access to manipulate the LEDs
* It is very important that you structure your code cleanly for this assignment.
* You must name all functions and adapt any menu help instructions appropriately.
* You will have to determine how the watchdog timeout duration is calculated. (This may be useful to research)
* On starting the game, ensure that all lights are first turned off.

## Demonstration

Please note that you must be prepared before offering to demonstrate your code. You must demonstrate the code that you have submitted to the eConestoga dropbox.

For this assignment, you will demonstrate the following items. You may use a lab PC or your own. If needed, you may use a classmate’s VM, but you MUST run your own code and only your own code. You must demonstrate in your regularly scheduled lab period. If you demonstrate modified code, you will receive a late penalty according to how long it has been since the dropbox deadline.

1. Your xx\_hook.c and xx\_asm.s code as taken directly from the drop box compiles.
2. Your code deploys with sudo make program.
3. Your code runs with no parameters provided (sensible defaults).
4. Your code runs as specified.
5. You can demonstrate both cases – that it is possible to both win and lose.

## Code

Your code will be evaluated as follows. Each will be evaluated on a 0 to 5 basis, where 0 is unacceptable or not present, and 5 is excellent.

### Functionality

Your program behaves as specified.

### Direct Memory Access

You have implemented all LED interaction as specified

### Watchdog

The watchdog works as specified.

### Proper use of Registers, the stack, and link register

Only necessary items (no more, no less) are pushed onto the stack. Your function calls use the link register appropriately.

### Code Readability and Structure

Your code is properly indented, does not wrap or have long lines, and can be logically understood.

You use subroutines and memory appropriately.

### Code comments

As this is assembly language, you must extensively comment any code that is unclear (i.e. nearly every line).

## Document Requirements

This assignment requires a single flowchart of the flow of your code. This flowchart should use standard flowchart shapes for steps and decisions, and should correspond with your logic.

Please print the table below, with your name clearly displayed, for marking purposes.

|  |  |
| --- | --- |
| Item | Grade |
| Demonstration | / 5 |
| Code |  |
| Functionality | / 5 |
| Direct Memory Access | / 5 |
| Watchdog | / 5 |
| Proper use of registers, etc. | / 5 |
| Readability | / 5 |
| Comments | / 5 |
| Code Subtotal | / 35 |
| Total | / 35 |
| Applicable Cap: |  |

### Caps

If you do not complete all functionality, the following caps will be applied:

|  |  |
| --- | --- |
| Function | Maximum Grade |
| Hard coded delays | 60% |
| Direct LED access missing | 80% |
| Watchdog demo missing | 80% |
| All functions implemented | 100% |

## Submission Guidelines

Please read and understand the submission guidelines below. Failure to correctly submit any required component will result in 10% penalty per missing or incorrect component.

### Paper Submission

Please print your flowchart and include the grade table. Ensure your name is on all pages.

### Electronic Submission

Exactly THREE items will be submitted electronically.

* A PDF of your flowchart
* Upload the complete C hook file xx\_hook.c.
* Upload your complete assembly file (the xx\_asm.s file).

For this assignment, please do not compress EITHER file, and please be certain you have named it and your functions correctly.

Complete file naming and upload guidelines are provided below.

# Electronic Submission and File Naming

Often, multiple groups will name their assignment files assignment1.zip (or some other common combination) which results in confusion and complications when downloading and unzipping submitted electronic files.

Please follow the following guidelines for all submissions this term. Failure to follow the electronic submission guidelines may result in a 10% penalty per infraction.

As long as other file naming conventions are followed, you may choose to append an assignment identifier (i.e. “\_a1” or something else as appropriate) as the last element before the file extension.

## Document Naming and Submission

Unless otherwise requested, you should only ever submit a single document containing the complete document as specified. This document must be:

* A professionally formatted document adhering to the SET report standard
* A single PDF file, not included in a zip

Unless indicated otherwise, do not submit the assignment specification as a part of your document.

Documents should be named with the usernames of all team members as follows:

* Individual assignments: username.pdf (i.e. jsmith.pdf)
* When working alone: username\_alone.pdf (i.e. jsmith\_alone.pdf)
* When working with a partner: username1\_username2.pdf (i.e. jsmith\_bjones.pdf)
* When working as a team: username\_team.pdf (i.e. jsmith\_team.pdf – but be certain to include all team members on report title page)