Lab Sheet 2: Polling and Interrupts

1 Aims

The aims of this lab are to:

- Understand how to use components on the shield (rather than only the development board).
- Understand how to use GPIO for polled input.
- Understand how to use a GPIO interrupt handler

1.1 Overview of Activities

Perform the following tasks and answer the questions on the answer sheet.

1. Activity 1: Set-up the sample project (accept assignment, clone repository). Download and run the project provided and answer questions about the given code
2. Activity 2: Implement a specified design to flash the LEDs on the shield and control them using button B1, which is polled.
3. Activity 3 [Advanced]: Implement a specified design to further elaborate the control of the LEDs using button B5, which triggers an interrupt.

Assessment. If you wish, you can present work on this lab exercise for assessment. Complete any of the following steps (see QM+ for full details and deadlines)

- **Answer the questions on the QMPlus quiz.** *You are recommended to look at the questions about the given code as you go along as answering them will help you complete the rest of the lab.*
- **Demonstration in the lab.** Ask one of the demonstrators to see your code working during the scheduled lab times.
- **Code assessment.** Complete the QM+ assignment to indicate that your code is to be assessed, provided you have demonstrated it. Do not attach any documents. The only way to submit code is to update your repository on GitHub. You can push code as often as you wish, whether or not you would like it assessed.

2 Activity 1: Download the Sample Project, Read the Code

A sample project is provided for this lab. The behaviour of the sample project is:

- Two LEDs on the shield are controlled using two buttons, B1 and B5.
- LED1 turns on or off (i.e. toggles) when button B1 is pressed.
- LED2 turns on or off (i.e. toggles) when button B5 is pressed.

2.1 Download, Read and Run the Sample Project

Read the provided code carefully. The project includes:

- Multiple files contain C code. If you are unsure how a C program can be written using multiple files, you are recommended to review the C slides / recording.
- Files `led.h` and `led.c` contain code for configuring and controlling the LEDs 1 to 5 on the shield.
- Files `button.h` and `button.c` have code for using the shield buttons B1 to B5.
- File `main.c` contains three tasks and the interrupt handler. **Only this files should be changed.**

2.2 Questions About the Given Code

The questions in the QMPlus quiz are about the given code. You may wish to look at these questions now.

3 Activity 2: Control LEDs using Button B1 and Polling

In this activity the code provided is elaborated. A design is given and you must implement this design.

3.1 Requirements

The requirements are:

1. One of the five LEDs 1 to 5 is lit at all times.
2. Every 1 sec, the lit LEDs moves, going from LED 1 to LED 2, LED 2 to LED3 etc and LED 5 back to LED 1. This is shown in Figure 1.
3. When button B1 is pressed, the direction of travel reverses so that when the 1 sec is completed, the change is in the opposite direction. This is shown in Figure 2.

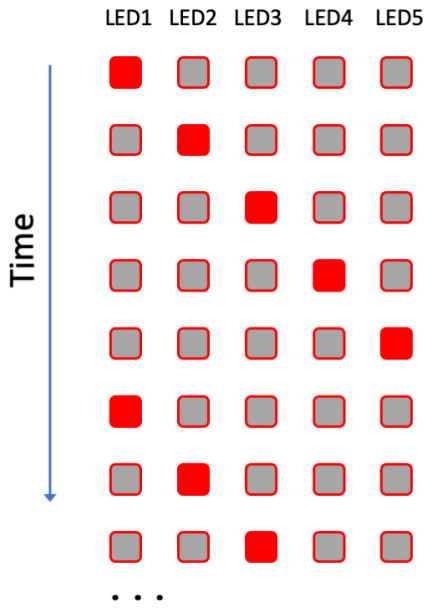

Figure 1: One lit LED moving from LED1 towards LED5

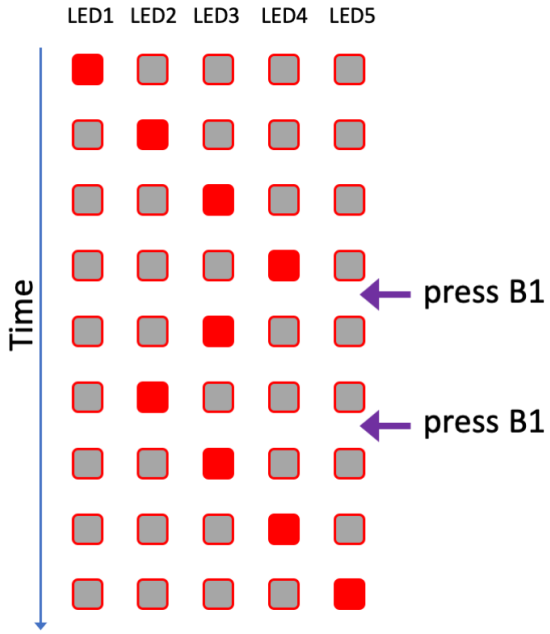

Figure 2: Direction of Movement Reverses

3.2 Design

There are many ways these requirements could be implemented. You are required to follow the following design.

Figure 3 shows a state transition diagram with two states, giving the direction of change. Implement this state system. The variable 'c' is a counter (you can give it a longer name). Do not use separate states for the 5 different patterns. Instead, an integer can be used to record which LED is lit. Turn off all the LEDs except this one. In Figure 3, this is represented by the function '`nextLit()`' which sets the next pattern of LEDs with one LED on, taking account of the direction. The function updates the integer variables used to record which LED is lt.

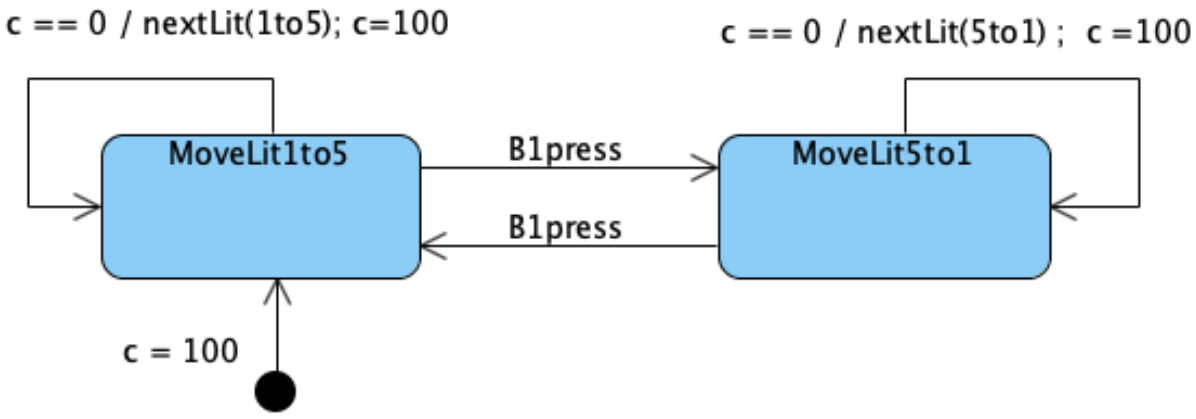

Figure 3: State Transition Diagram 1

4 Activity 3: Control LEDs using Button B5 and an Interrupt [Advanced]

In this activity the code written for activity 2 is elaborated.

4.1 Requirements

The additional requirements are:

1. Pressing button B5 causes the LEDs to invert, so that all the LEDs are lit except the one that was previously lit. This is shown in Figure 4.
2. This change should occur immediately (i.e. on the next cycle of the cyclic system). The unlit LED then moves when the remainder of the 1 sec interval (shown as T s in Figure 4) has elapsed.
3. The direction of moving does not change but now the single unlit LED moves.

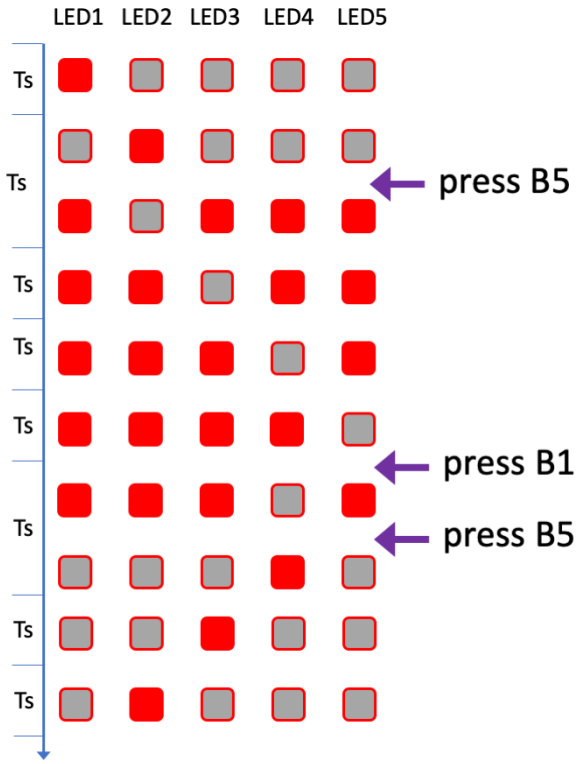

Figure 4: Invert and Reverse the Moving LED Pattern

4.2 Design

The required design is shown in Figure 5, two extra states have been added: in both of the new states all the LEDs except 1 are ON, so that the one that is OFF moves. There is a state for each direction.

Again, separate states are not used for the different patterns as an integer can be used.

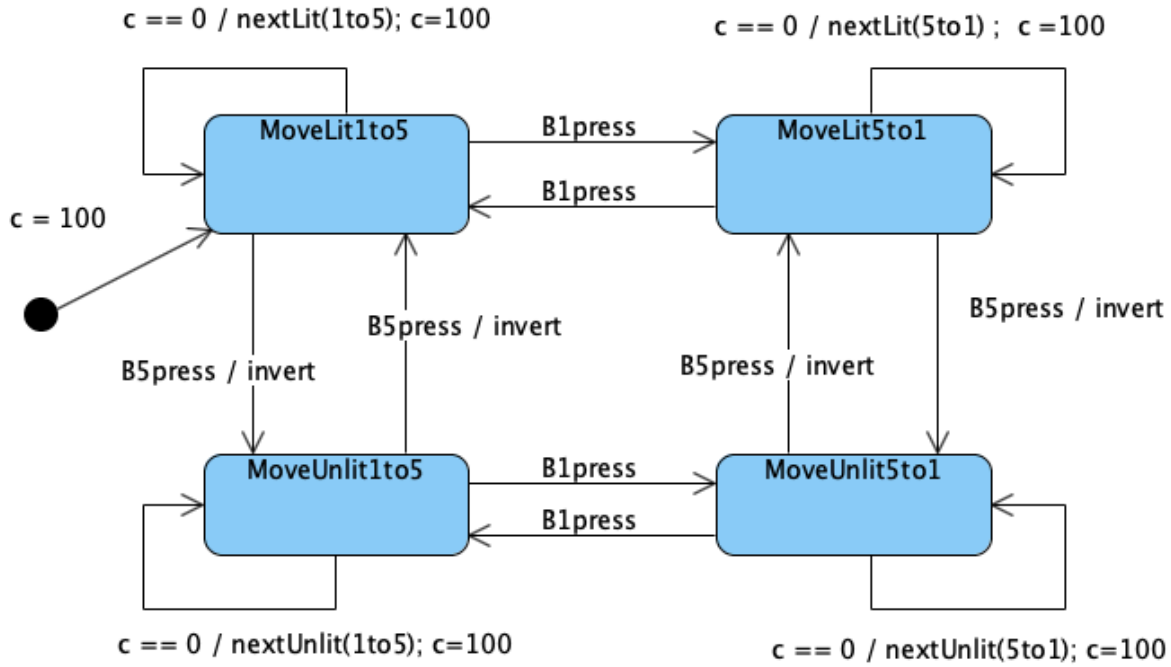

Figure 5: State Transition Diagram 2

Checklist for presentation and coding

1. Presentation points to check for:
   - Check the formatting of code (watch out for tabs)
   - Ensure that all code that is not used has been removed.
   - Ensure that the README is updated so that it describes your work.
   - Ensure that the code is commented and all the comments are relevant and correct.
2. **Compliance with the coding / design guidelines.** (See https://qmplus.qmul.ac.uk/mod/page/view.php?id=1396803)
   - The guidelines for a cyclic system apply here