Chapter 9
Low-Level Hardware Drivers in C++

Microcontroller applications usually require low-level drivers for peripheral devices such as I/O ports, interrupts, timers, communication interfaces like UART, CAN [4], SPI™, etc. This chapter presents several efficient methods for programming peripheral hardware drivers in C++. Low-level drivers are inherently dependent on the microcontroller and its peripherals. Even though the low-level hardware drivers in this chapter are primarily designed for our target with the 8-bit microcontroller, an effort has been made to keep them as portable as possible. In this way, they can be adapted to other microcontrollers. The final section in this chapter presents a complete, non-trivial example of controlling a seven-segment display [3].

9.1 An I/O Port Pin Driver Template Class

General purpose I/O ports can be used for a variety of interfaces to on-board and off-board devices. A simple general purpose I/O port can be controlled via three registers, an output data register for setting the output value, a direction register for selecting input or output, and an input data register for reading the input value. Port pins often come grouped in registers that are 8, 16 or 32 bits wide. The general purpose I/O ports on our target with the 8-bit microcontroller, for example, can be controlled with three 8-bit registers, the data register, the direction register and the input register, as shown in Table 9.1.

We will now write a template port_pin class that encapsulates a port pin in one of the general purpose I/O ports, portb, portc or portd, as summarized in Table 9.1.
Table 9.1 The registers of the general purpose I/O ports on our target with the 8-bit microcontroller are summarized

<table>
<thead>
<tr>
<th>I/O port</th>
<th>Data register</th>
<th>Direction register</th>
<th>Input register</th>
</tr>
</thead>
<tbody>
<tr>
<td>portb</td>
<td>0x25</td>
<td>0x24</td>
<td>0x23</td>
</tr>
<tr>
<td>portc</td>
<td>0x28</td>
<td>0x27</td>
<td>0x26</td>
</tr>
<tr>
<td>portd</td>
<td>0x2B</td>
<td>0x2A</td>
<td>0x29</td>
</tr>
</tbody>
</table>

template<typename addr_type,
           typename reg_type,
           const addr_type port,
           const reg_type bpos>
class port_pin
{
  public:
    static void set_direction_output()
    {
      // Set the port pin direction to output.
      port_dir_type::bit_set();
    }

    static void set_direction_input()
    {
      // Set the port pin direction to input.
      port_dir_type::bit_clr();
    }

    static void set_pin_high()
    {
      // Set the port output value to high.
      port_pin_type::bit_set();
    }

    static void set_pin_low()
    {
      // Set the port output value to low.
      port_pin_type::bit_clr();
    }

    static bool read_input_value()
    {
      // Read the port input value.
      port_inp_type::bit_get();
    }
}