1 Introduction

This document serves as a reference on how to assemble the elevator used in the elevator project. Table 1 summarizes the pinout. In the following sections the peripherals are described in more detail.

1.1 Push Buttons and LEDs

Four of the five push buttons are used to call the elevator to the respective floor. The fifth button serves as an emergency stop button. They are connected to the STM32 using 10 kΩ pulldown resistors.

220 Ω resistors are used to connect the LEDs. The yellow LED signals that the cabin has stopped at a floor. The red LED lights up after pushing the emergency stop button and is switched back off after pushing the stop button once again.

1.2 Distance Measurement

The distance from the cabin to the ground is measured using the ultrasonic ranging module HC-SR04\(^1\). Distance values are obtained from the module by following these steps:

1. Trigger a measurement by applying a high level signal to the trigger pin for at least 10 µs.
2. Measure the duration \(T_{\text{echo}}\) of the received echo at the echo pin.
3. The distance \(d\) can be calculated using the velocity of sound (340 m s\(^{-1}\)): \(d = 0.5 \cdot T_{\text{echo}} \cdot v_{\text{sound}}\).

1.2.1 Using TIM8 for Distance Measurement on STM32

The most sensible way to drive the ranging module from an STM32 is to use one of the timer IPs to generate a periodic trigger pulse (using PWM mode of one channel), and then using “input capture” on another channel pair of the same timer to record the pulse length. The STM32 connected to the elevator has timer TIM8 chosen for this purpose. Note that these pins are also used for SD card communication, so any SD card must be removed from the board before TIM8 is used to drive a ranging module.

Cube Configuration

Aside from the pin configurations from table 1, you also need to configure the TIM8 IC. In the IC configuration list under the “Pinout” tab, for “Channel1” select “PWM Generation CH1”, for “Channel3” select “Input Capture direct mode”, and for “Channel4” select “Input Capture indirect mode” (this configures Channel4 to do input capture on the Channel3 pin).

Under the “Configuration” tab, click “TIM8” which should now have appeared under “Control”. Set “Prescaler” to “96-1”. Since the timer is driven by a 96 MHz clock (c.f. “APB2 timer clocks” under the “Clock Configuration”

\(^1\)The datasheet of the HC-SR04 is available online at \url{http://www.micropik.com/PDF/HCSR04.pdf}.

<table>
<thead>
<tr>
<th>STM32 Pin</th>
<th>Cube Config</th>
<th>Header Pin</th>
<th>Peripheral</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC8</td>
<td>TIM8.CH3</td>
<td>CN3-18</td>
<td>Ultrasonic Sensor — Echo Pin</td>
</tr>
<tr>
<td>PC6</td>
<td>TIM8.CH1</td>
<td>CN3-21</td>
<td>Ultrasonic Sensor — Trigger Pin</td>
</tr>
<tr>
<td>PA1</td>
<td>TIM2.CH2</td>
<td>CN1-36</td>
<td>H-Bridge — Enable Channel 1 and 2</td>
</tr>
<tr>
<td>PC2</td>
<td>GPIO.Output</td>
<td>CN1-32</td>
<td>H-Bridge — Channel 1 Input</td>
</tr>
<tr>
<td>PC12</td>
<td>GPIO.Output</td>
<td>CN4-34</td>
<td>H-Bridge — Channel 2 Input</td>
</tr>
</tbody>
</table>

Table 1: The pinout of the STM32.
tab), this makes TIM8 tick every microsecond. Set “Counter Period” to “25000-1” to make the period 25 ms (fastest operating period of the ranging module). Scroll down to “PWM Generation Channel 1” and set “Pulse” to “20”, for 20µs pulses. Under “Input Capture Channel 4”, set “Polarity Selection” to “Falling Edge”. Now the last rising edge of the ECHO pin will be recorded in the Channel3 capture register, and the last falling edge in the Channel4 capture register. Under the “NVIC Settings” tab, enable the “capture compare” interrupt.

**Reading the Timer from C**  First you need to enable the timer. Do the following at startup, which starts the PWM and Input Capture, and enables the interrupt.

```c
HAL_TIM_PWM_Start(&htim8, TIM_CHANNEL_1);
HAL_TIM_IC_Start(&htim8, TIM_CHANNEL_3);
HAL_TIM_IC_Start_IT(&htim8, TIM_CHANNEL_4);
```

The measured distances are best read from the timer interrupt. In order to handle it, define

```c
void HAL_TIM_IC_CaptureCallback (TIM_HandleTypeDef *tim)
```

which is called when the timer expires. If `tim == &htim8`, then TIM8 generated the interrupt. You can read the contents of the capture registers with

```c
int time = HAL_TIM_ReadCapturedValue(&htim8, TIM_CHANNEL_3);
```

And similarly for `TIM_CHANNEL_4`.

### 1.3 DC Motor

The cabin is moved using a 12 V DC motor[2]. Its speed is adjusted by varying the applied voltage. There is a threshold voltage that must be exceeded to move the cabin. The threshold voltage depends on the load. The rotation direction of the DC motor depends on the polarity of the applied voltage.

Two channels of an H-Bridge[3] are used to control the motor from the Arduino Uno. The basic operations are:

1. Channel 1 high and channel 2 low: Rotation in one direction.
2. Channel 1 low and channel 2 high: Rotation in the other direction.
3. To vary the speed, apply a PWM signal to the enable pin of Channel 1 and 2.

---