Part 1.1

The assembly program implements and controls the LED light panel to map the situation of slider switches directly to LED panel. It used an infinite loop to read slider switches and then write their situation to the LED panel's memory address.

Part 1.2

(This subpart was not finished after a few attempts due to heavy load of other courses)

Part 2.1

The assembly program implements a up-counter from 1 to 15 (represented in e). It uses an infinite loop to regularly check if the private timer has a TimeOut set to 1. If so, it updates the register that is used for the counter and then update the HEX Display accordingly.

Part 2.2

The assembly program implements a stopwatch using the idea of polling. It uses an infinite loop to check if any pushbuttons are pushed and released, by checking the EdgeCapture of the pushbuttons. If so, it will conduct actions like STOP, START, and RESET. The timer part uses an infinite loop to update the registers used for counting milliseconds, seconds, as well as minutes, according to the EdgeCapture of Pushbuttons, and then update the HEX Display accordingly.

Part 3. The assembly program implements a stopwatch that is similar to the program in part 2.2. It uses an interrupt-based approach instead. It uses the generic arm interrupt handler to update tim_int_flag every time when the private timer triggers an interrupt. The infinite main loop (IDLE) then reads and checks the tim_int_flag to update registers used for counting the time. When pushbuttons generate an interrupt, they will update PB_int_flag, which will also be read during an infinite loop and used to decide what actions (START, STOP, RESET) shall be taken for the stopwatch.