BAME3104 REAL-TIME OPERATING SYSTEMS

Q1. Buttman & Rawbean Sdn. Bhd. is an embedded security systems solution provider. Their current project is to develop the firmware for the electronic door lock system, known as the *i* Diot. The security door lock system comes with a numeric keypad which allows a user to key in the code in order to unlock, two LEDs with different colors as indicators, and a buzzer as notifier, as shown in Figure 1.1.

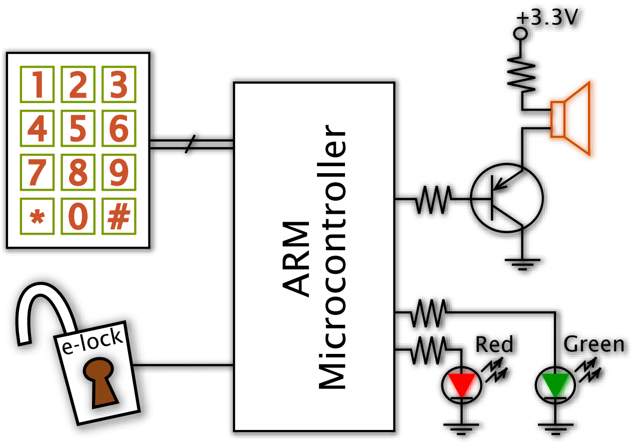

Figure 1.1: *Door Lock System*

In the normal operating state, the door is locked, the buzzer is off, the red LED is on, but the green LED is off. To unlock, a user has to enter the 4 correct digits number of which were preprogrammed in the flash memory of the μ-controller. On each key press, the buzzer beeps to notify the user of a key press. When the code entered is verified to be valid, the system waits for half a second before unlocking. If another key is pressed during this time, it is assumed that it was a fluke of luck that the correct code was entered the first time. In this case, the buzzer generates a tone to indicate an invalid code for 2 seconds. On the other hand, if all goes well the door is unlocked for 5 seconds, the green and red LEDs are turned on and off, respectively, for the same period of time, and the buzzer goes off for 2 seconds to notify that the door has been unlocked. After that, the system goes back to its normal state.

Draw the UML State Machine for *i* Diot and use the following provided C API to facilitate your design and implementation. You can create any required variables, but you must clearly state what they are used for. Please take some time to read the annotated functions thoroughly to be clear of their roles and behaviors.

```
typedef enum {OFF, ON} LEDState;
typedef enum {UNLOCK, LOCK} LockCommand;
typedef enum {NO_TONE, KEYPRESS_TONE, VALID_CODE_TONE,
              INVALID_CODE_TONE} Tone;

/**
 * Get the user key-press (if any). The function does not block, which
 * means it will always return even though no user presses a key.
 *
 * Return the key-press if any, otherwise return 0xff
 */
char getKeyPress(void);
```

Q1. (Continued)

```
/**
 * Turn on or off the green LED
 *
 * Input:
 *    state  OFF means turn off
 *           ON means turn on
 */
void turnGreenLED(LEDState state);

/**
 * Same as turnGreenLED() except that it affects the red LED
 */
void turnRedLED(LEDState state);

/**
 * Send a signal to door controller to lock or unlock the door.
 *
 * Input:
 *    cmd    UNLOCK means unlock the door
 *           LOCK means lock the door
 */
void signalDoor(LockCommand cmd);

/**
 * Send a signal to buzzer controller to generate a tone. The
 * function does not block, which means it returns immediately.
 * The buzzer controller stops the buzzer automatically when it
 * is done.
 *
 * Input:
 *    tone   NO_TONE means shut off the buzzer
 *           KEYPRESS_TONE means generate key-press tone for 1 sec
 *           VALID_CODE_TONE means generate valid tone for 2 sec
 *           INVALID_CODE_TONE means generate invalid tone for 2 sec
 */
void signalBuzzer(Tone tone);

/**
 * Return current time in microseconds
 */
unsigned int getTime(void);
```

(50 marks)
[Total: 50 marks]