**General description:**

The project is developed through several stages.

1. The project is split into small parts to be implemented.
2. Each of the part is assigned to one of the members.
3. Members complete their own part and test their code on the board.
4. Codes are integrated into a whole, then general test is applied.

MS Teams meeting and WeChat voice call are used to exchange idea of completing the project.

Contributions of each member are included in the file ‘works distribution’ and are as followed:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Work | Description | Need test | Deadline | Owner |
| Overall design | 1. Hardware components and related pins connections.  2. Software components, interfacing.  3. Variable(register) convention. | No | 2021/11/03 | Xiaohang Hu |
| LED brightness control | Control 2 groups of LEDs brightness by PWM.  4 levels:  0-clear  1-light opaque  2-medium opaque  3-dark | Yes | 2021/11/14 | Boqin Huang |
| Emergency control | · Push button triggers 4 LEDs flashing through interrupt.  · Push again would stop flashing also through interrupt. | Yes | 2021/11/14 | Sixiang Qiu |
| LCD display state | LCD display state information as required. | Yes | 2021/11/14 | Zheyuan Shao |
| Integrate, general test | Integrate the code with keypad scanning part. | Yes | 2021/11/18 | Xiaohang Hu |
| Demonstration | Test Camera rack. | No | 2021/11/18 | Raymond Ye |
| Doublecheck, board test | Give tests to each work and the whole one. | Yes | 2021/11/18 | Raymond Ye |

**Overview of the project design:**

**Hardware components:**

**LEDs:**

* Window1: LED0-LED1
  + Pins connections:
    - PL3(OC5A, 16 bits, timer5) to LED0.
    - PL4(OC5B, 16 bits, timer5) to LED1.
* Window2: LED3-LED4
  + Pins connections:
    - PE3(OC3A, 16 bits, timer3) to LED3.
    - PE5(OC3C, 16 bits, timer3) to LED4.
      * Certain reason causes PE4 not working.
* Emergency flashes: LED6-LED9
  + Pins connection:
    - PG0-PG3 to LED6-LED9.

**Keypad:**

* Individual control:
  + Key 1: window 1 decreasing
  + Key 4: window 1 increasing
  + Key 2: window 2 decreasing
  + Key 5: window 2 increasing
    - decreasing: window brighter = LED darker.
    - increasing: window darker = LED brighter.
* Central control:
  + A: set to 0-clear.
  + B: set to 3-dark.
  + C: exit central control. window back to local state.
* Pins connection:
  + PC0 To C3
  + PC1 To C2
  + PC2 To C1
  + PC3 To C0
  + PC4 To R3
  + PC5 To R2
  + PC6 To R1
  + PC7 To R0

**Push button:**

* PB0 for Emergency control.
* Use interrupt.
* Pins connection:
  + PD0(INT0) to PB0

**LCD:**

* Display state information.
* Pins connection:
  + PF0-PF7 to D0-D7
  + PA5 to RW
  + PA6 to E
  + PA7 to RS

**Software components:**

**Code structure:**

1. Comment of authority and interfacing.
2. Definition of registers:

* Used when scanning the keypad.
* r15 - r19
* Prevents duplicate interrupts.
* r20 (interrupt\_flag)
* Used as parameters, function variables and return values.
* r21 - r26
* Determines the status depending on user input:

|  |  |  |
| --- | --- | --- |
| **Register** | **Explanation** | **Values** |
| r0: control\_state | the state of simulation | 0: Initial state(S)  1: Local control (L)  2: Central control(C)  3: Emergency (!!!) |
| r1: emergency | If it is in emergency mode | 0: not in emergency mode  1: in emergency mode |
| r2: win1\_level | window 1 opacity level | 0: clear  1: light opaque  2: medium opaque  3: dark |
| r3: win2\_level | window 2 opacity level | 0: clear  1: light opaque  2: medium opaque  3: dark |

1. Definition of constant value.
2. Interrupt functions:

* OpO\_interrupt(push button)
* set ‘emergency’ register.
* if ‘emergency’ set, clear ‘emergency’, restore previous status.
* RESET

1. Main function:

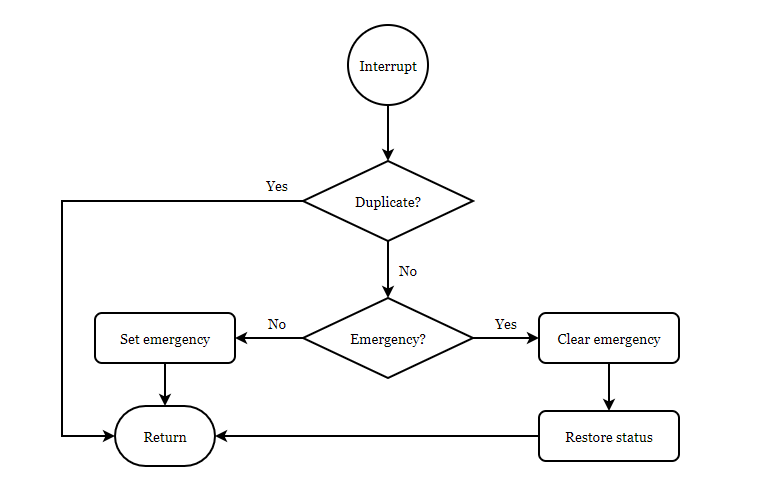
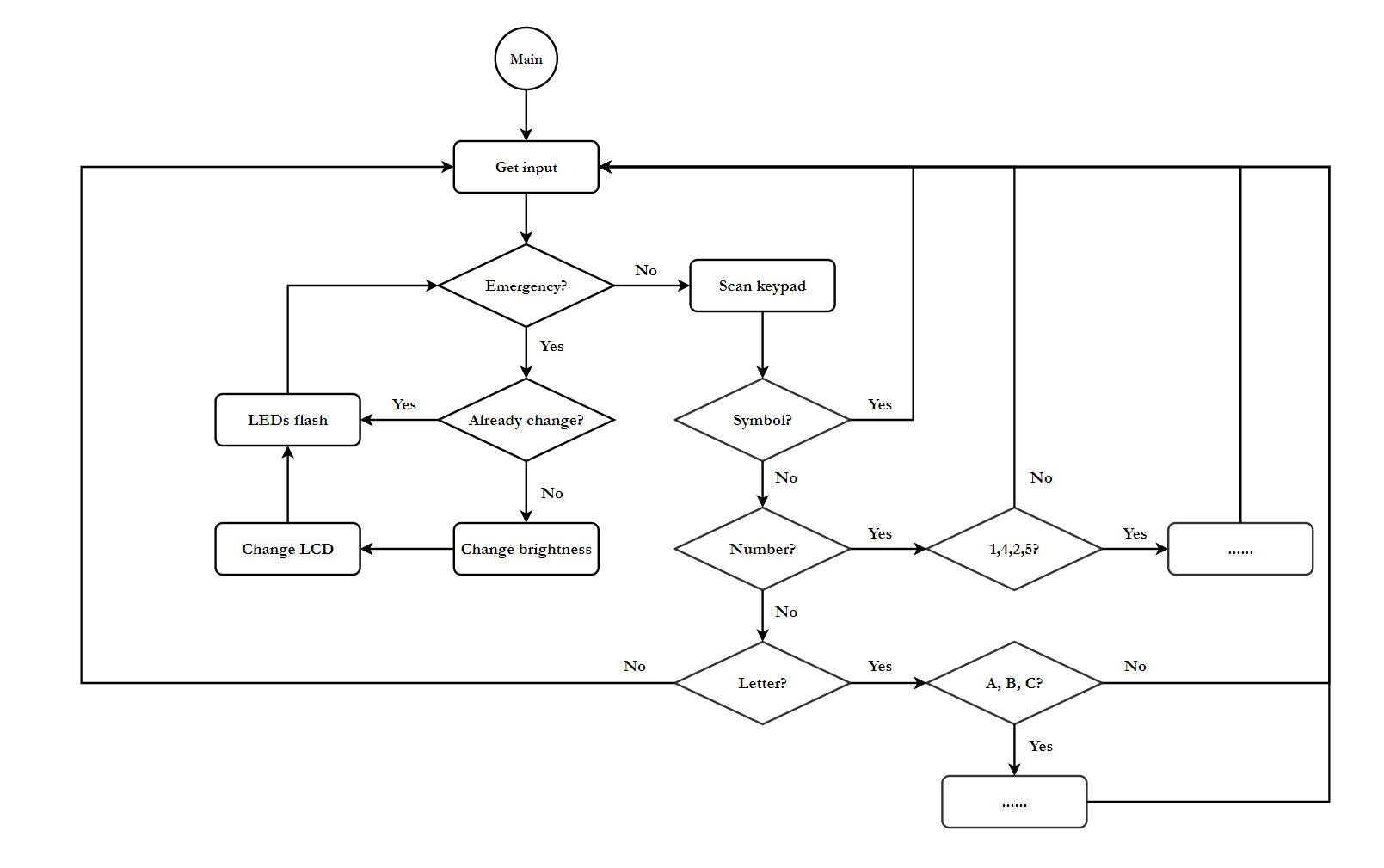
* get\_input
* If ‘emergency’ set:

1. set\_window\_brightness, LCD\_display\_state.(see below)
2. endless loop(LEDs flash).

* break and return if ‘emergency’ cleared.
* if ‘emergency’ cleared , scan keypad(get\_input\_do).
* Other utility functions.(see below)

**Keypad component:**

* All key push actions follow the priority requirement.
* Key 1:
  + condition: control\_state = 0 or 1
  + decreasing the window1 opacity level
  + decreasing win1\_level, bottom to 0
  + set control\_state to 1
* Key 4:
  + condition: control\_state = 0 or 1
  + increasing the window1 opacity level
  + increasing win1\_level, up to 3
  + set control\_state to 1
* Key 2:
  + condition: control\_state = 0 or 1
  + decreasing the window2 opacity level
  + decreasing win2\_level, bottom to 0
  + set control\_state to 1
* Key 5:
  + condition: control\_state = 0 or 1
  + increasing the window2 opacity level
  + increasing win2\_level, up to 3
  + set control\_state to 1
* Key A:
  + condition: control\_state = 0 or 1 or 2
  + setting all windows to clear
    - set win1\_level and win2\_level to 0
  + set control\_state to Central control (2)
* Key B:
  + condition: control\_state = 0 or 1 or 2
  + setting all windows to dark
    - set win1\_level and win2\_level to 3
  + set control\_state to Central control (2)
* Key C:
  + condition: control\_state = 2
  + Exit central control
  + set control\_state to Local control (1)
* Invoke following functions after setting the value of registers.
* set\_windows\_brightness:
* set the brightness of both windows given their status(corresponding registers’ value) using PWM technique.
* LCD\_display\_states:
* display the status of both windows on LCD.
* Then, go back to main function.

**Execution flow:**

**Conclusive remarks:**

The project is completed by joint efforts of all Group 3 members. Cheerfully, the outcome almost meets all of the project requirements.

During the work, every aspect covered in the lectures is put into practice. Registers are defined following rules in the documentation. Various functions are implemented based on instructions over those registers.

* Delay function
* set\_window\_brightness function
* LCD\_display\_state function

Meanwhile, input/output devices are connected with codes by PORT. The interfacing rules are declared in the documentation of the project as well.

* Push button, keypad...
* LCD/LEDs, use of PWM technique...

Finally, interrupt is introduced to hold certain events(in this case, activating the emergency mode and quitting from it).

But still, the final workout has shortages in specific areas. Important thing is that the design of the project lacks concern on simultaneous input issue. Despite that there is only 1ms delay for each loop of scanning the keypad, simultaneous input should be taken into account. One of the solutions might be queueing the input and set timer to deal with the queue which is identical to the concept of parallel input.

Sixiang Qiu