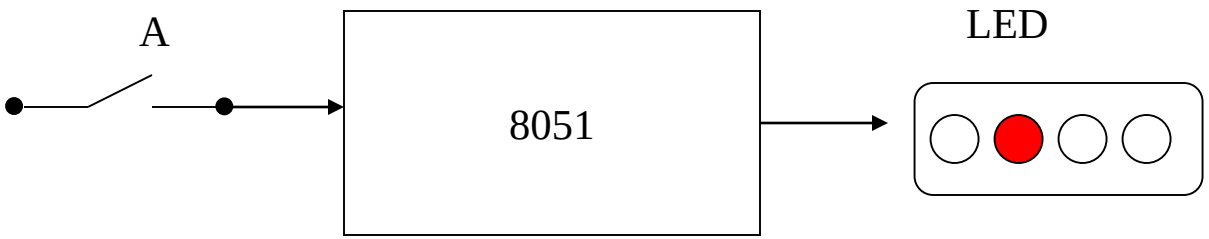
Lab 04

Button Control and De-Bounce Filter

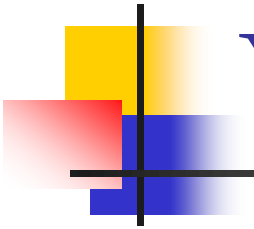


Your Task

- Control the LED display by pressing a button



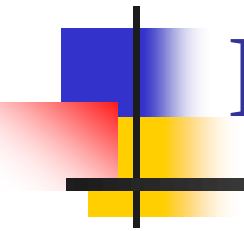
Things You Need to Know for Your Task



(1) How to program 8051 to receive an input signal from the button?

(2) How to filter-out unstable signal when a button pressed?

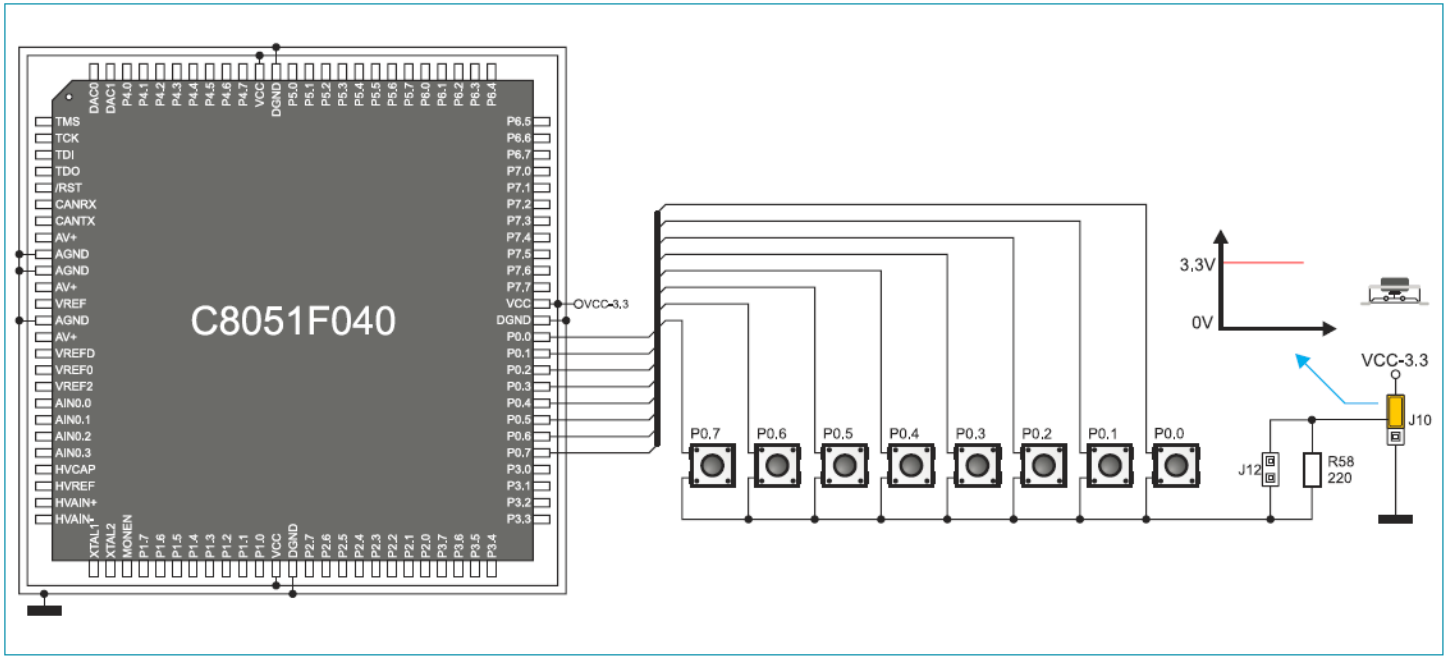
- The de-bounce filter

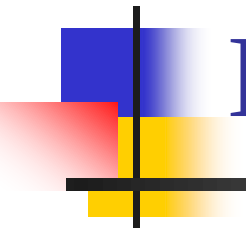


How to Receive a Button Input

How to Detect the Push Button

- A hit generates a logic 1 to an GPIO pin

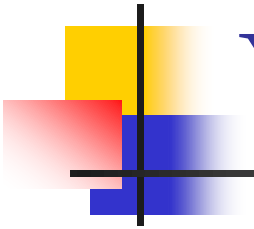




How to Count the Number of Hits to a Button?

Deal with unstable signal

Things You Need to Know for Your Task



(1) How to program 8051 to receive an input signal from the button?

(2) How to filter-out unstable signal when a button pressed?

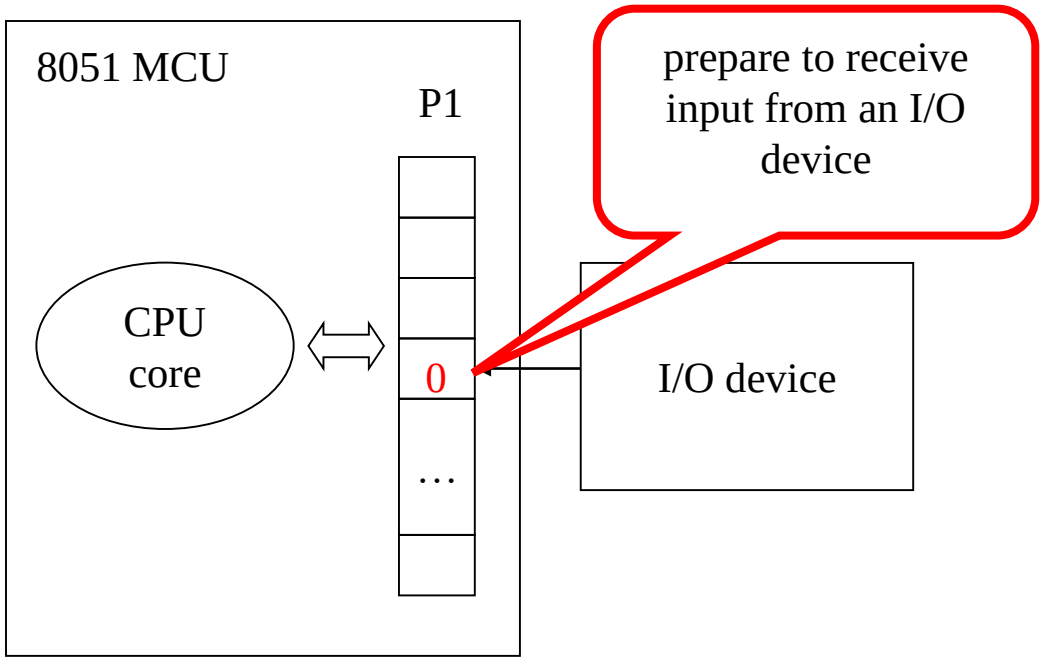
- The de-bounce filter

You May Write Such a Program

```
count = 0;

while (1) {
      while (P1.3==0);

      //goto here if P1.3==1
      count++;
}
```

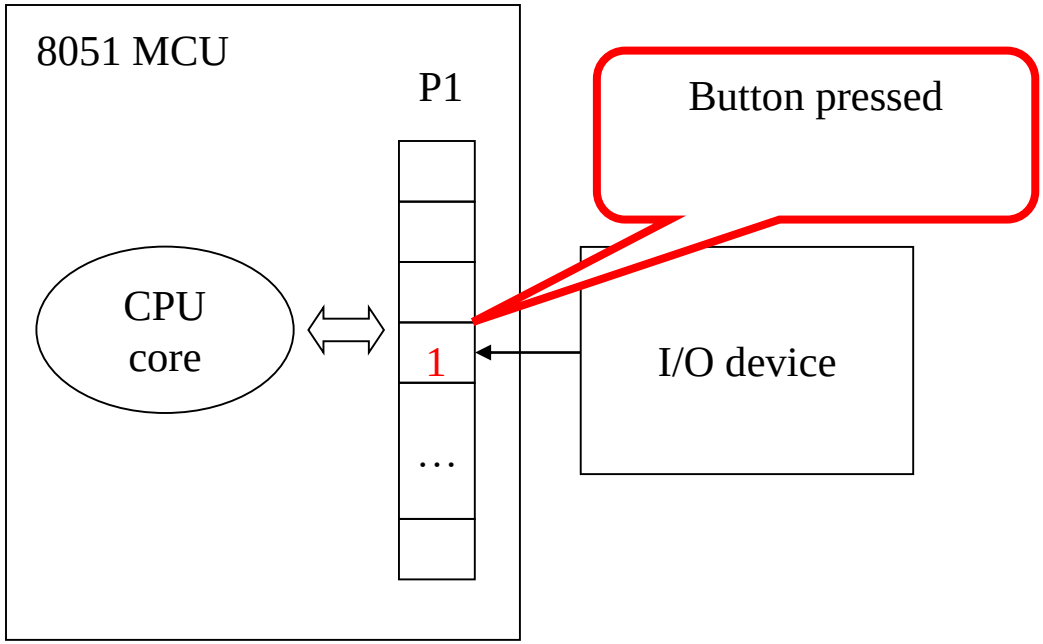


You May Write Such a Program

```
count = 0;

while (1) {
        while (P1.3==0);

        //goto here if P1.3==1
        count++;
}
```



You May Write Such a Program

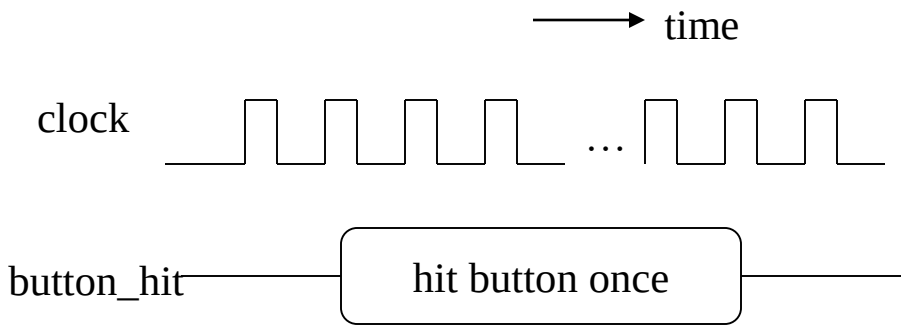
- What's wrong with this program?

```
count = 0;

while (1) {
        while (P1.3==0);

        //goto here if P1.3==1
        count++;
}
```

(1) You think you just hit the button once but the CPU sense it for hundreds of times



You May Write Such a Program

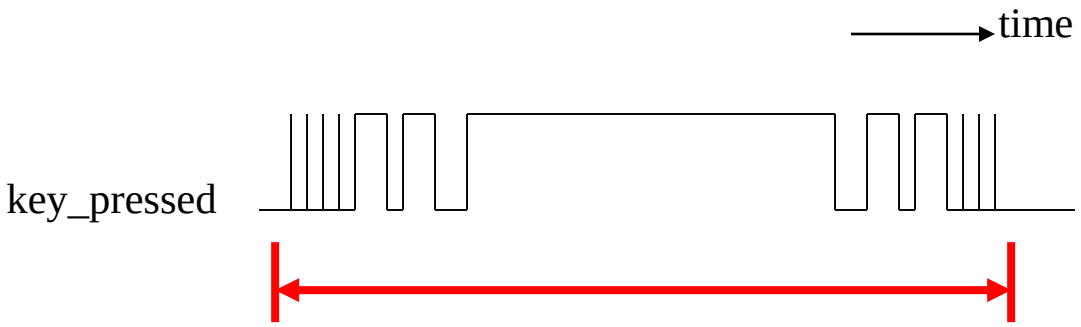
- What's wrong with this program?

```
count = 0;

while (1) {
        while (P1.3==0);

        //goto here if P1.3==1
        count++;
}
```

(2) Unstable signal when you pressed a button



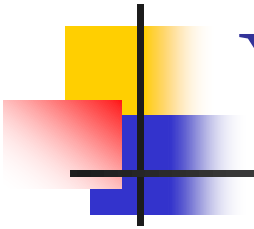
you pressed just once
but generate lots of pulses



De-bounce Filter

The design concepts

Things You Need to Know for Your Task

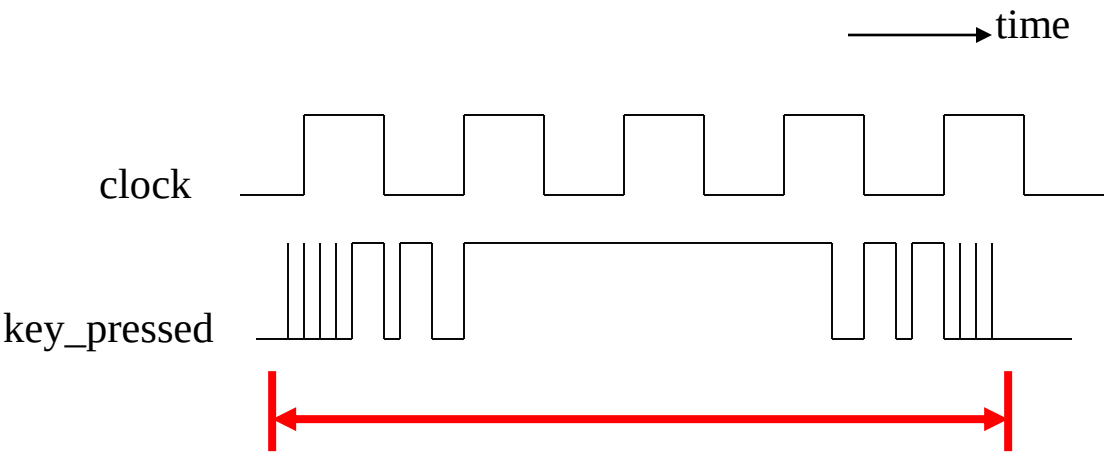


(1) How to program 8051 to receive an input signal from the button?

(2) How to filter-out unstable signal when a button pressed?

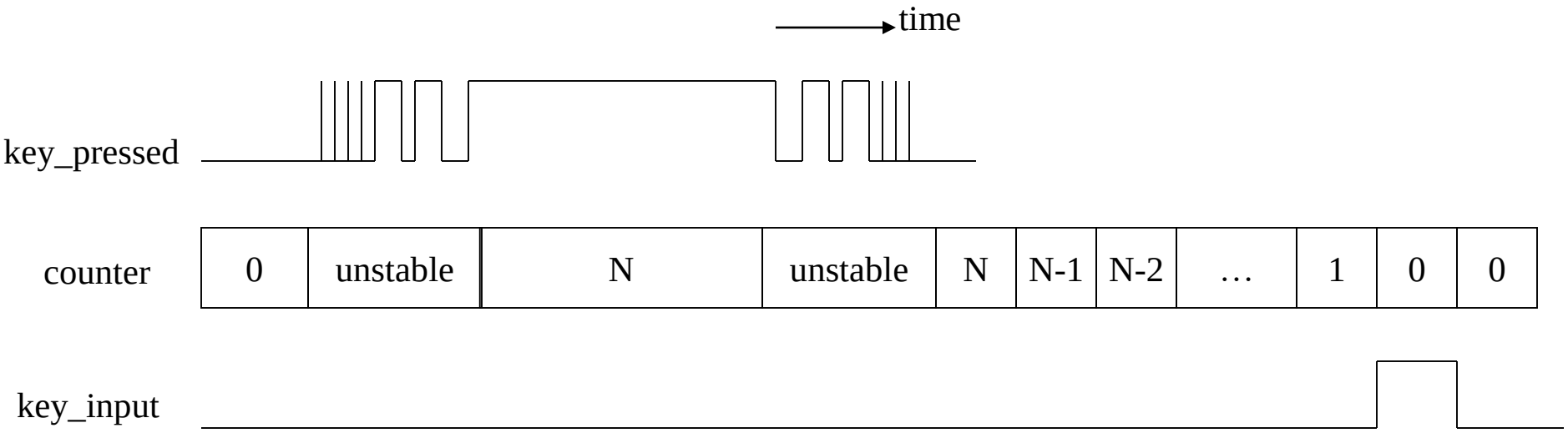
- The de-bounce filter

How to Filter-out Multiple Pulses from a Press



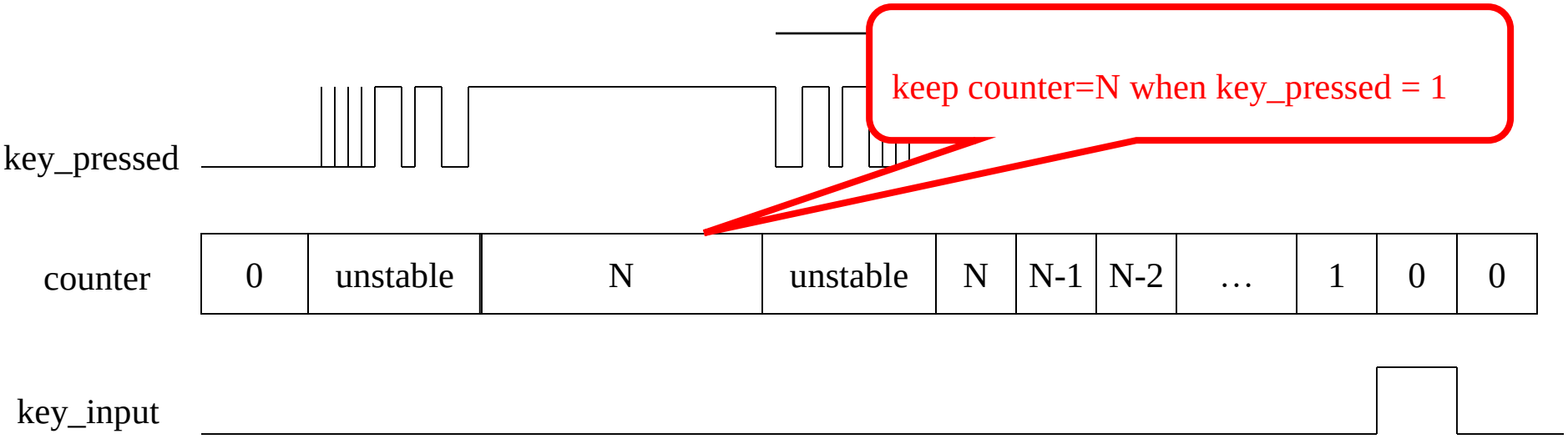
How to Filter-out Multiple Pulses from a Press

- Use a counter to count the time to stable



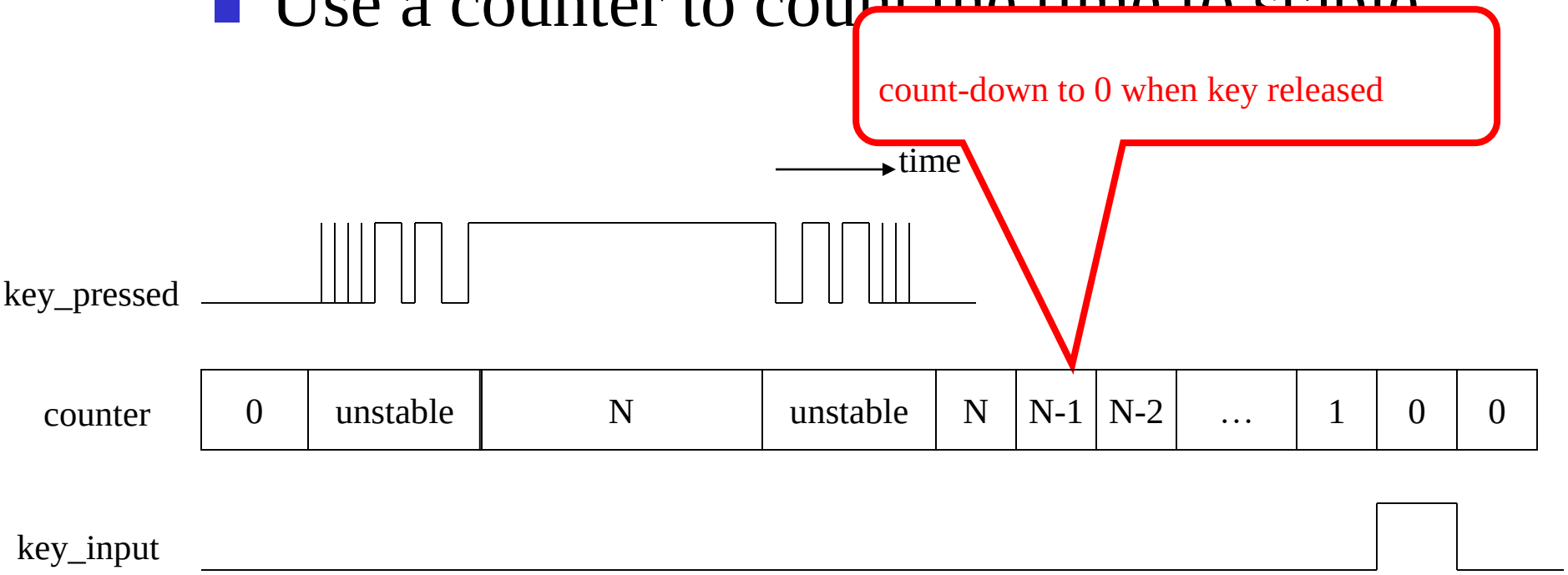
How to Filter-out Multiple Pulses from a Press

- Use a counter to count the time to stable



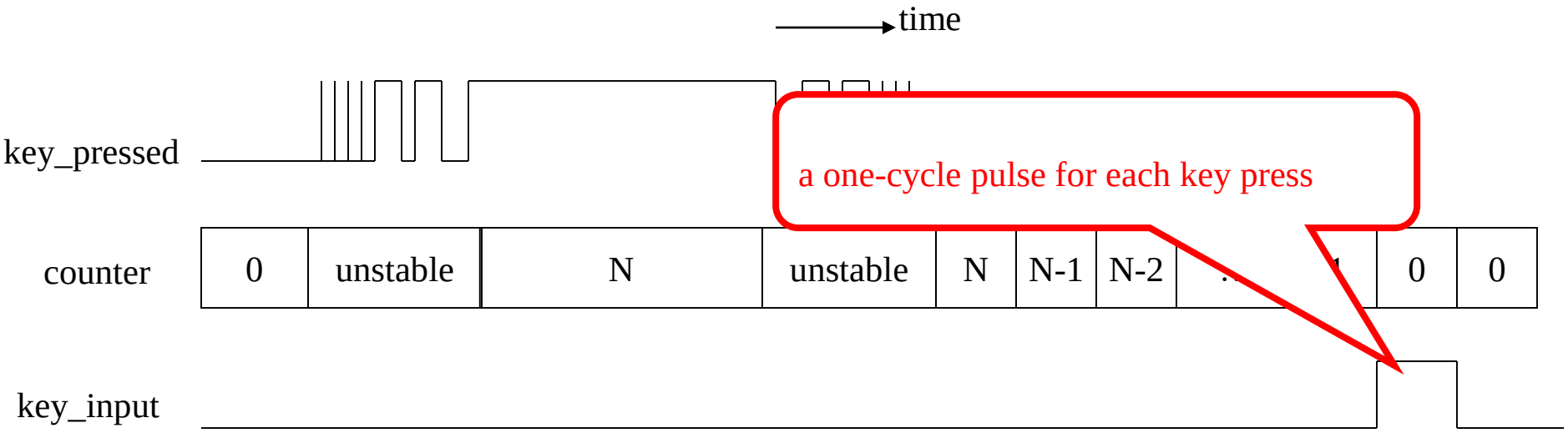
How to Filter-out Multiple Pulses from a Press

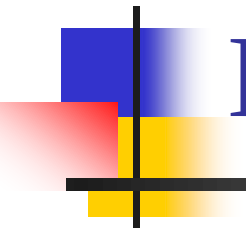
- Use a counter to count the time to stable



How to Filter-out Multiple Pulses from a Press

- Use a counter to count the time to stable



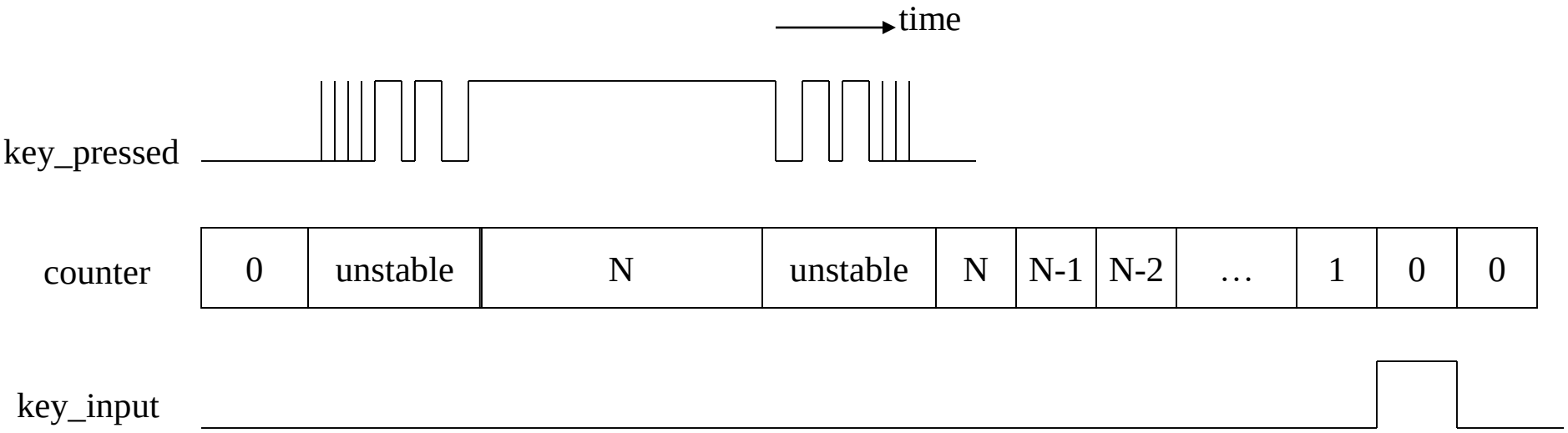


De-Bounce Filter

How to program

How to Filter-out Multiple Pulses from a Press

- Write a program for the hardware concept

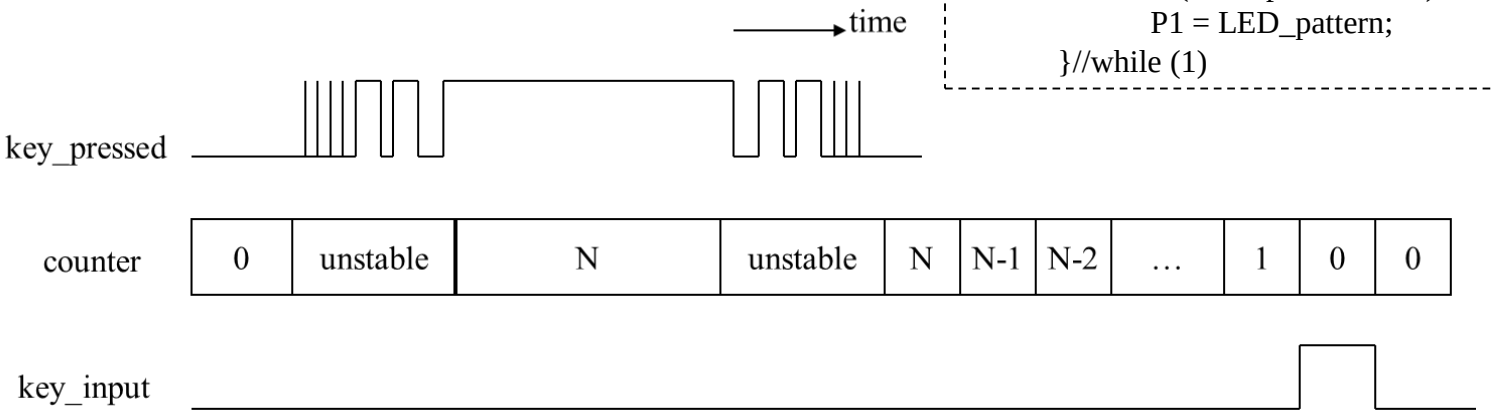


De-Bounce Filter Programming

```
while (1) {
        //Stage 1: wait for a button pressed
        do {
                key_hold = P0;
        } while (!key_hold);

        //Stage 2: wait for key released
        key_release = 0;
        count = N;
        while (!key_release) {
                key_hold = P0;
                if (key_hold) {
                        count = N;
                }
                else {
                        count--;
                        if (count==0) key_release = 1;
                }
        }//Stage 2: wait for key released

        //Stage 3:move LED pattern
        LED_pattern = (LED_pattern << 1)+1;
        if (LED_pattern=0xff) LED_pattern = 0xfe;
        P1 = LED_pattern;
}//while (1)
```

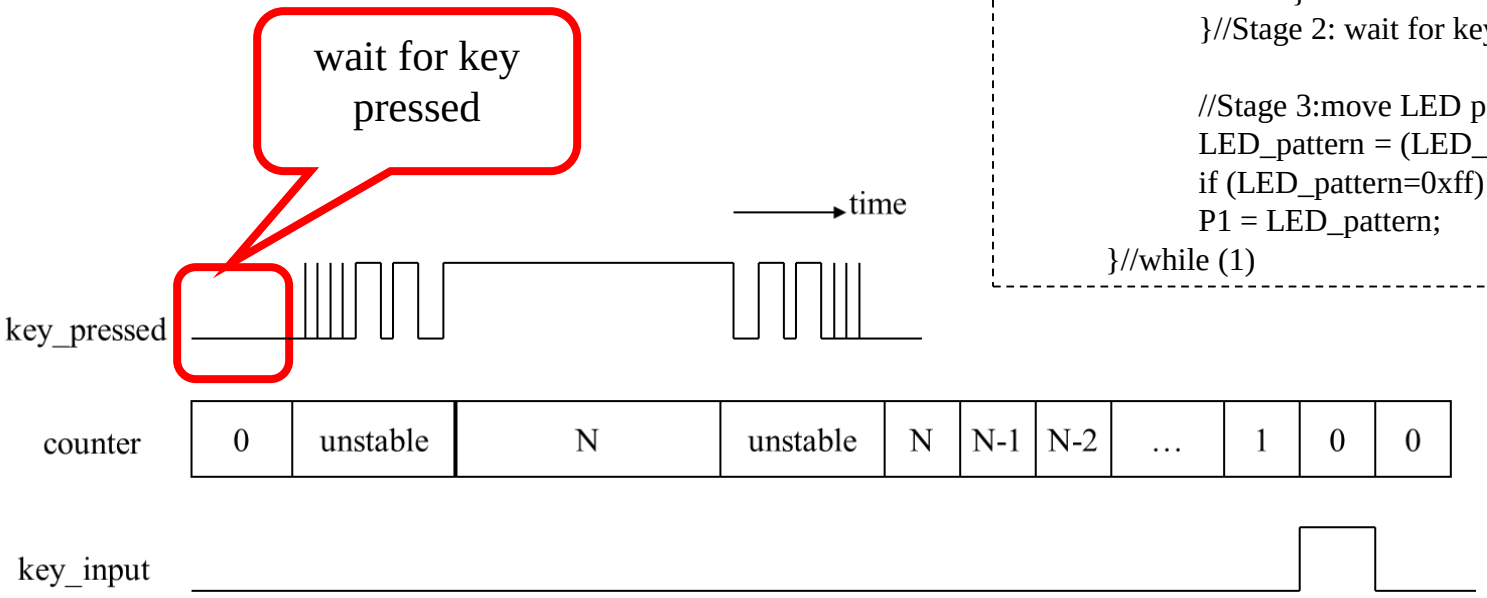


De-Bounce Filter Programming

```
while (1) {
        //Stage 1: wait for a button pressed
        do {
                key_hold = P0;
        } while (!key_hold);

        //Stage 2: wait for key released
        key_release = 0;
        count = N;
        while (!key_release) {
                key_hold = P0;
                if (key_hold) {
                        count = N;
                }
                else {
                        count--;
                        if (count==0) key_release = 1;
                }
        }//Stage 2: wait for key released

        //Stage 3:move LED pattern
        LED_pattern = (LED_pattern << 1)+1;
        if (LED_pattern=0xff) LED_pattern = 0xfe;
        P1 = LED_pattern;
}//while (1)
```



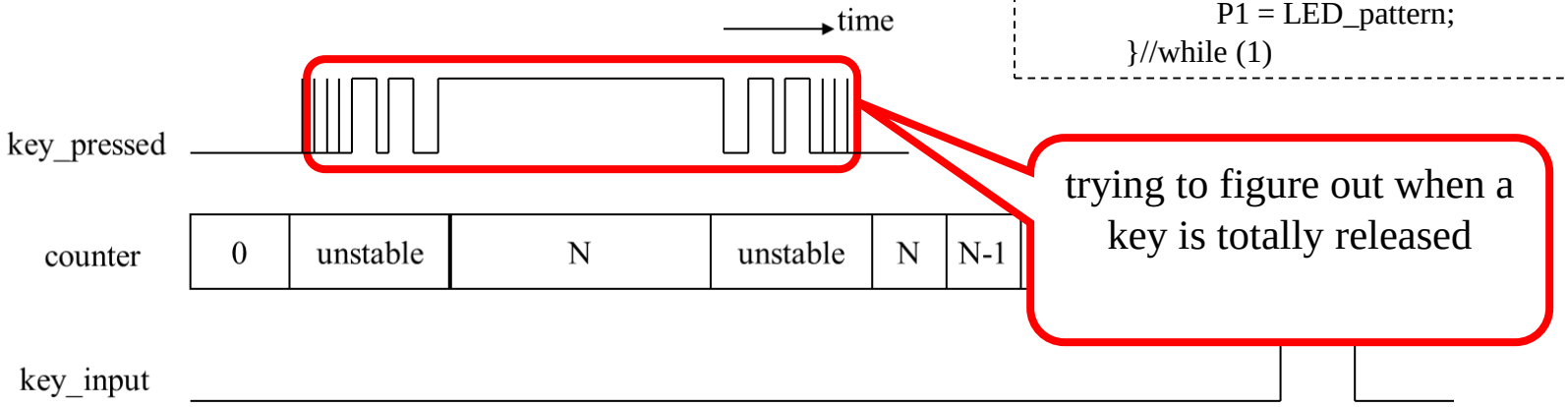
De-Bounce Filter Programming

- Keep counter=N when key hold
- Start count-down when key_hold=0

```
while (1) {
        //Stage 1: wait for a button pressed
        do {
                key_hold = P0;
        } while (!key_hold);

        //Stage 2: wait for key released
        key_release = 0;
        count = N;
        while (!key_release) {
                key_hold = P0;
                if (key_hold) {
                        count = N;
                }
                else {
                        count--;
                        if (count==0) key_release = 1;
                }
        }//Stage 2: wait for key released

        //Stage 3:move LED pattern
        LED_pattern = (LED_pattern << 1)+1;
        if (LED_pattern=0xff) LED_pattern = 0xfe;
        P1 = LED_pattern;
}//while (1)
```



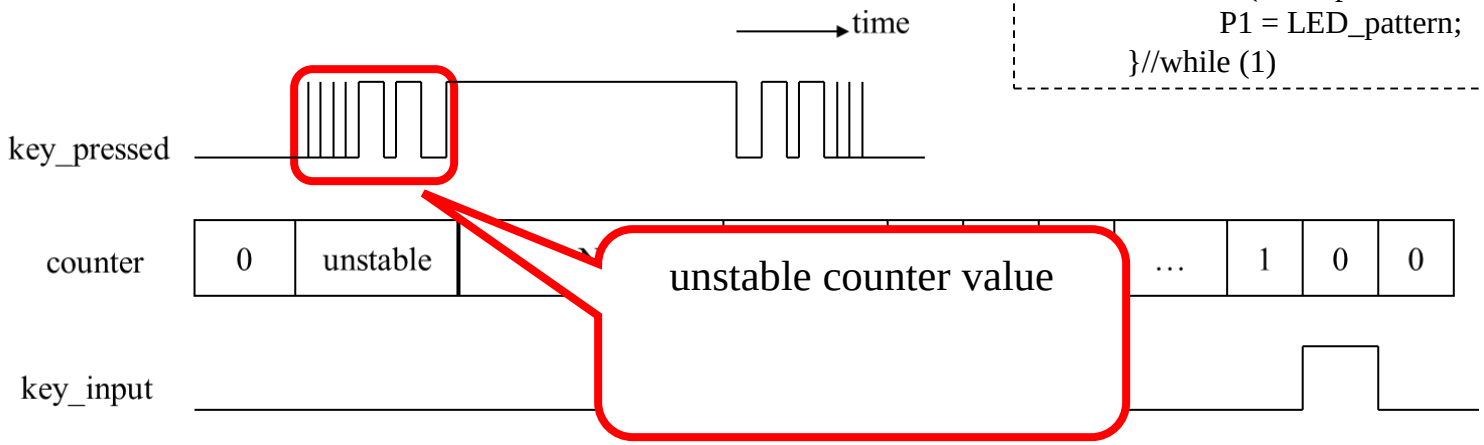
De-Bounce Filter Programming

- Keep counter=N when key hold
- Start count-down when key_hold=0

```
while (1) {
        //Stage 1: wait for a button pressed
        do {
                key_hold = P0;
        } while (!key_hold);

        //Stage 2: wait for key released
        key_release = 0;
        count = N;
        while (!key_release) {
                key_hold = P0;
                if (key_hold) {
                        count = N;
                }
                else {
                        count--;
                        if (count==0) key_release = 1;
                }
        }//Stage 2: wait for key released

        //Stage 3:move LED pattern
        LED_pattern = (LED_pattern << 1)+1;
        if (LED_pattern=0xff) LED_pattern = 0xfe;
        P1 = LED_pattern;
}//while (1)
```



De-Bounce Filter Programming

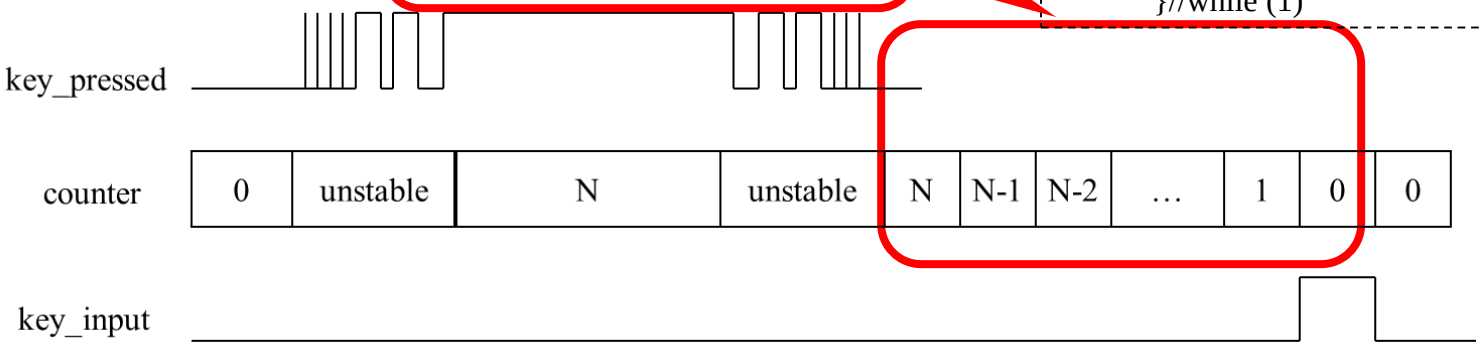
- keep counter=N when key hold
- start count-down when key_hold=0

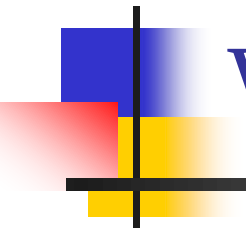
```
while (1) {
        //Stage 1: wait for a button pressed
        do {
                key_hold = P0;
        } while (!key_hold);

        //Stage 2: wait for key released
        key_release = 0;
        count = N;
        while (!key_release) {
                key_hold = P0;
                if (key_hold) {
                        count = N;
                }
                else {
                        count--;
                        if (count==0) key_release = 1;
                }
        }//Stage 2: wait for key released

        //Stage 3:move LED pattern
        LED_pattern = (LED_pattern << 1)+1;
        if (LED_pattern=0xff) LED_pattern = 0xfe;
        P1 = LED_pattern;
}//while (1)
```

count-down to 0 when a key is totally released





How to Make Two I/O Devices Work Simultaneously

About the bonus

You May Write Such a Program

```
while (1) {
        wait_button_pressed ();
        btn_count++;
        LED_display ();
}
```

You May Write Such a Program

```
while (1) {
        wait_button_pressed ();
        btn_count++;
        LED_display ();
}
```

```
do {
      key_hold = P0;
} while (!key_hold);

//Stage 2: wait for key released
key_release = 0;
count = N;
while (!key_release) {
      key_hold = P0;
      if (key_hold) {
            count = N;
      }
      else {
            count--;
            if (count==0) key_release = 1;
      }
}//Stage 2: wait for key released
```

You May Write Such a Program

```
while (1) {
        wait_button_pressed ();
        btn_count++;
        LED_display ();
}
```

```
//scan for each digit
for (i=0;i<3;i++) {
    P0 = pattern (digit[i]);
}
```

You May Write Such a Program

- What's wrong with this program?

```
while (1) {
        wait_button_pressed ();
        btn_count++;
        LED_display ();
}
```

```
//scan for each digit
for (i=0;i<3;i++) {
    P0 = pattern (digit[i]);
}
```

```
do {
        key_hold = ~P0;
} while (!key_hold);

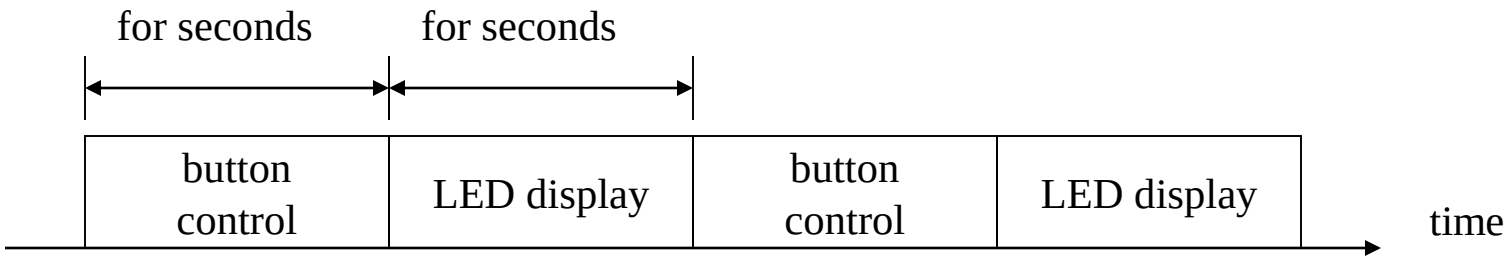
//Stage 2: wait for key released
key_release = 0;
count = N;
while (!key_release) {
        key_hold = ~P0;
        if (key_hold) {
                count = N;
        }
        else {
                count--;
                if (count==0) key_release = 1;
        }
}//Stage 2: wait for key released
```

You May Write Such a Program

- What's wrong with this program?

```
while (1) {
        wait_button_pressed ();
        btn_count++;
        LED_display ();
}
```

- You will never see button control and digit display work together

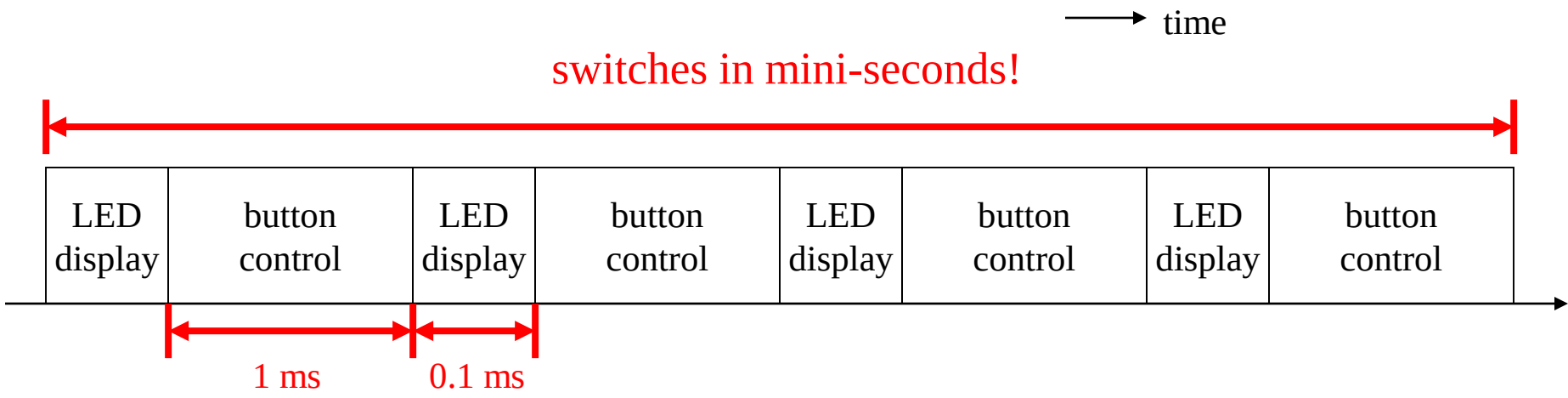




Time-Sharing to Control Multiple I/O Devices

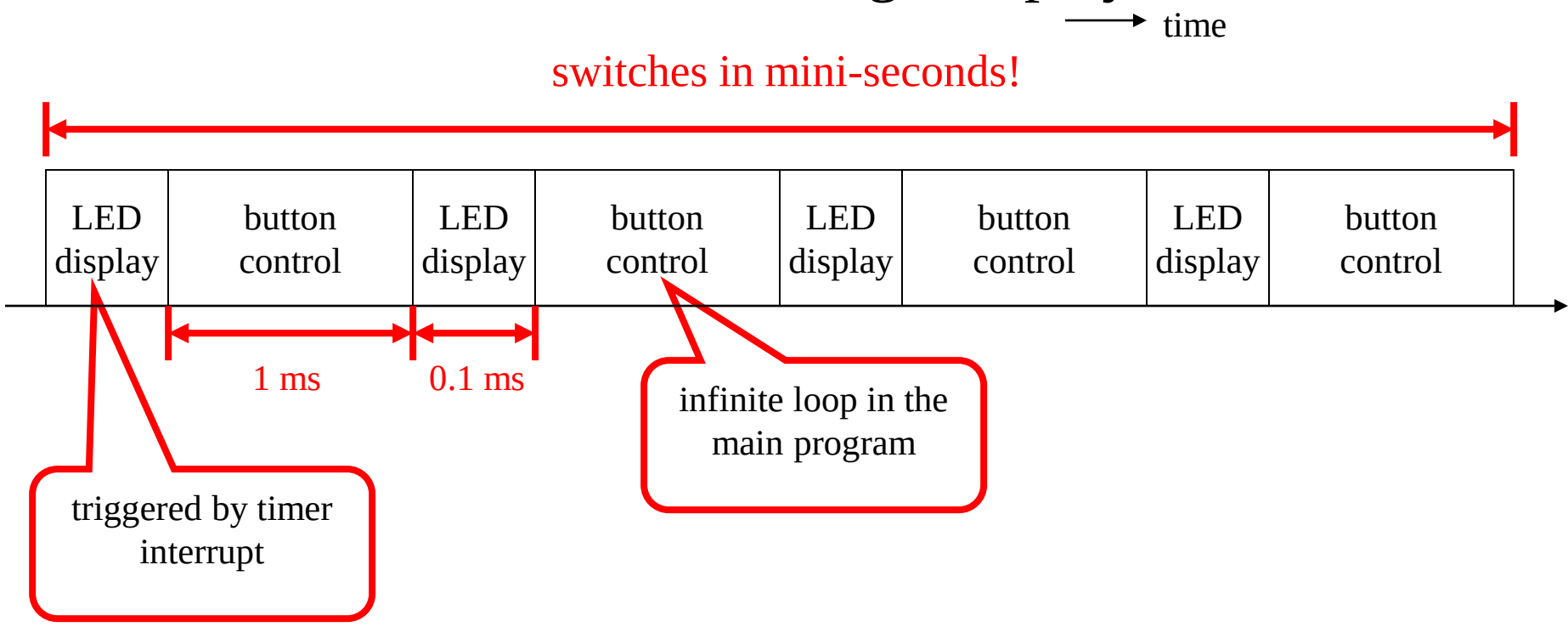
The Correct Scheme

- Time-sharing to control all the I/O devices



A Scheme for Time-Sharing Control

- Main program: for button control
- Timer ISR: to scan for digit display



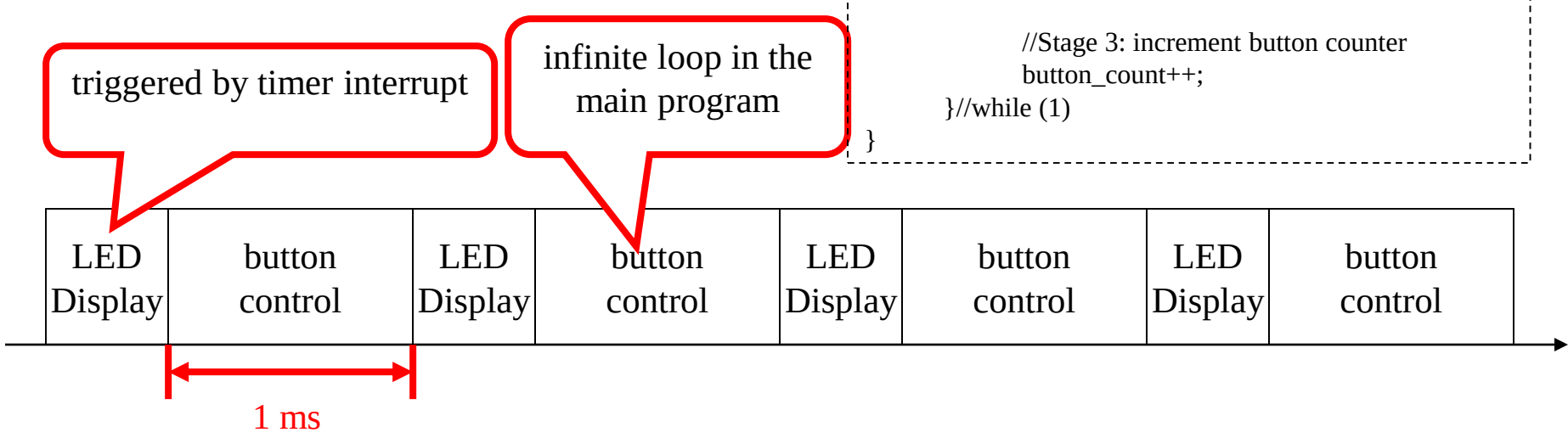
A Scheme for Time-Sharing Control

```
Timer_ISR () {
    switch LED pattern;
}
```

```
main () {
    ….
        while (1) {
            //Stage 1: wait for a button pressed
            do {
                key_hold = P0;
            } while (!key_hold);

            //Stage 2: wait for key released
            key_release = 0;
            count = N;
            while (!key_release) {
                key_hold = P0;
                if (key_hold) {
                    count = N;
                }
                else {
                    count--;
                    if (count==0) key_release = 1;
                }
            }//Stage 2: wait for key released

            //Stage 3: increment button counter
            button_count++;
        }//while (1)
}
```



Lab Requirements

- Basic Part:
  - Each hit of the button moves up/down the LED
  - one hit -> shift left, one hit -> shift right, one hit -> shift left, one hit -> shift right, …
- Bonus: (10 Points)
  - The LED runs automatically
    - Shift right or left per second
  - Each hit change the direction of the LED shifting

Lab04 Study Report

- File name: Bxxxxxxx-MCE-Lab4-Study
- File type: PDF only
- The requirements of report
  - Summarize the content of this slide set
  - Provide your plan for this lab exercise
  - No more than one A4 page
  - Grading: 80 ± 15
- Deadline: 2026/11/11 23:00 (不收遲交)
- Upload to e-learning system

Lab04 Lab Exercise Report

- File name: Bxxxxxxx-MCE-Lab4-Result
- File type: PDF only
- The requirements of report
  - Summarize the problems and results you have in this exercise
  - Some screen shots or some code explanation can be provided
  - No more than two A4 pages
  - Grading: 80 ± 15
- Deadline: 2026/11/18 23:00 (不收遲交)
- Upload to e-learning system