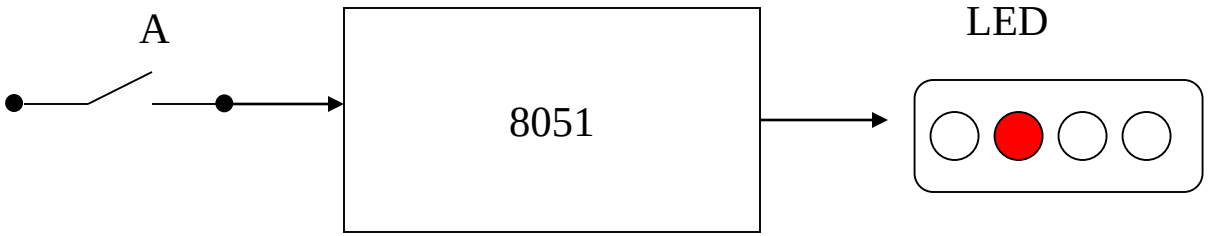
Lab 04

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

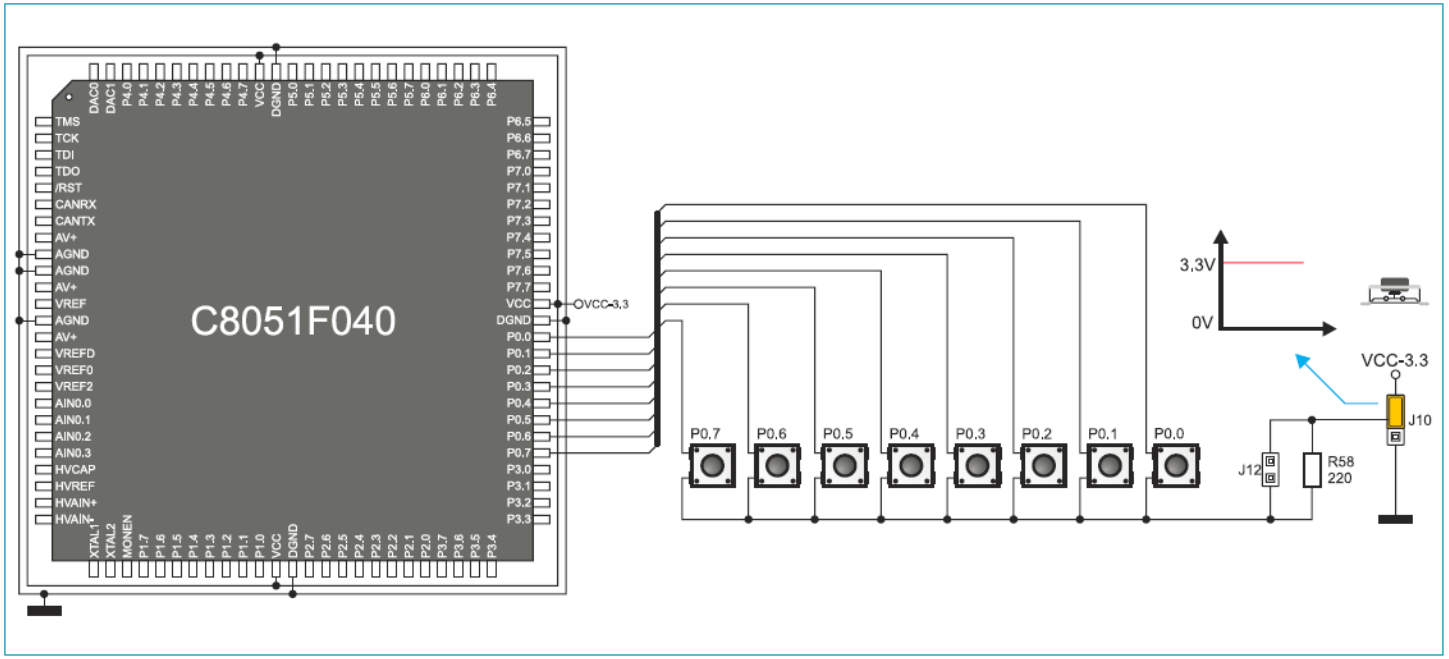


Figure 17-2: Push buttons and port PORT0 connection schematic

How to count number of hits to a button?

deal with unstable signal

Things you need to know for your task

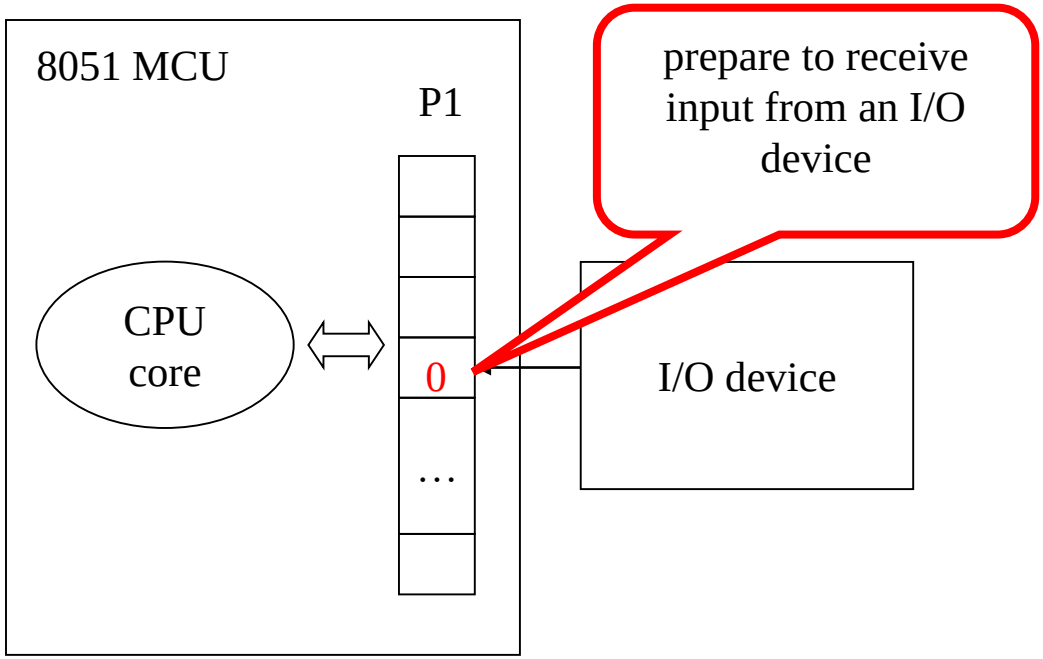
(1) How to program 8051 to receive an input signal from the button?

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

- the de-bounce filter

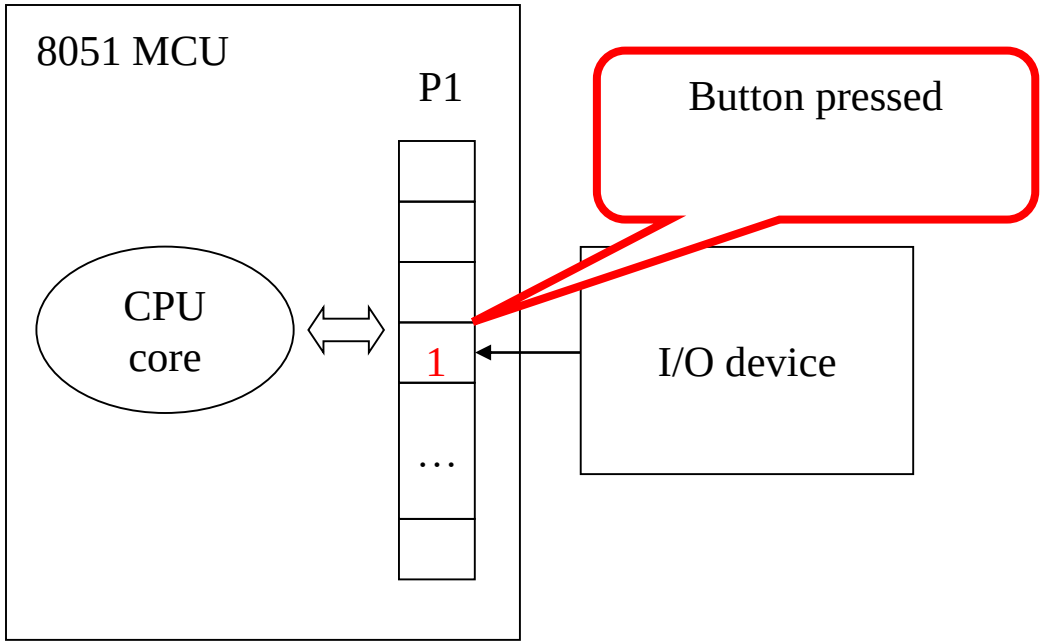
You may write such a program

```
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++;
}
```

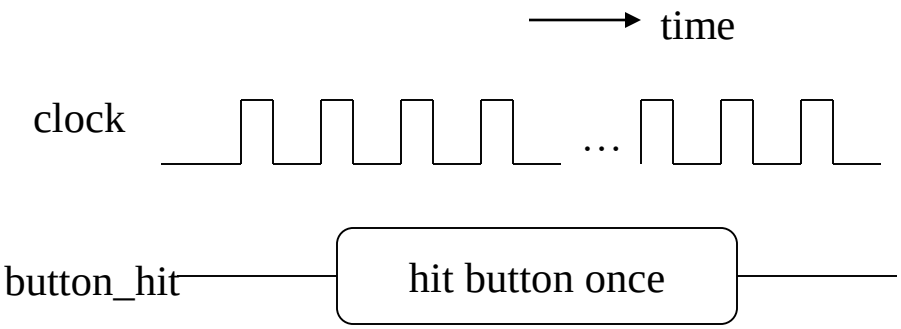


You may write such a program

- 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

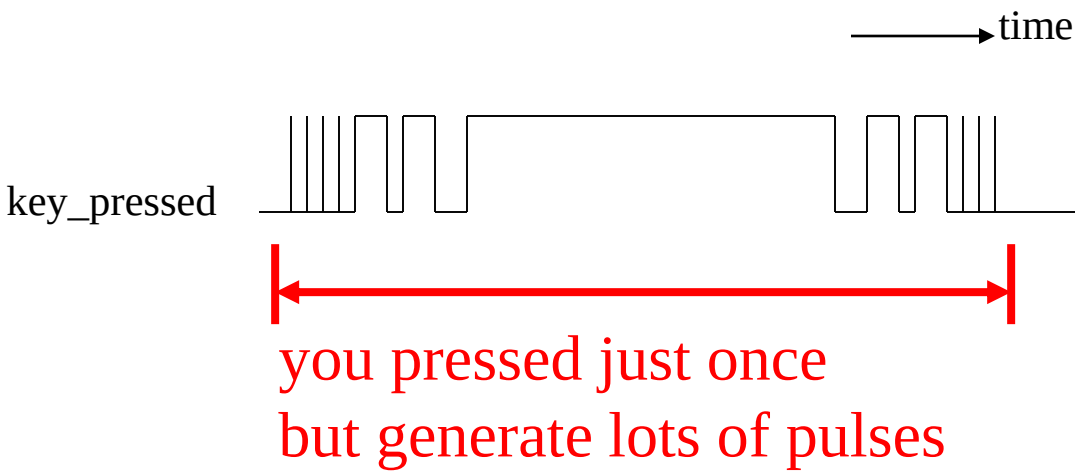


You may write such a program

- 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



De-bounce filter

the design concepts

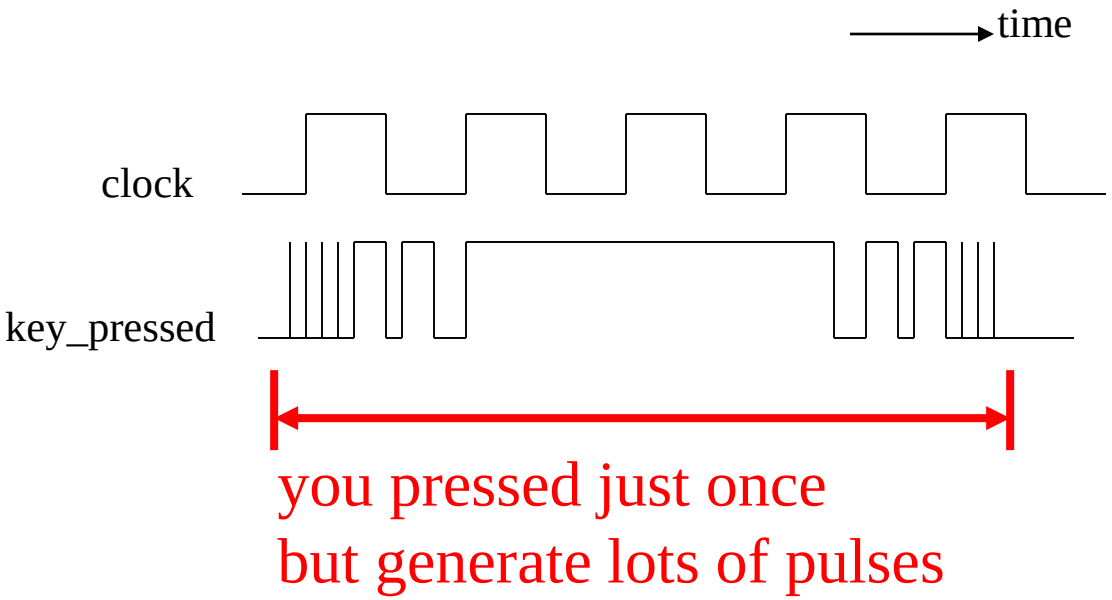
Things you need to know for your task

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

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

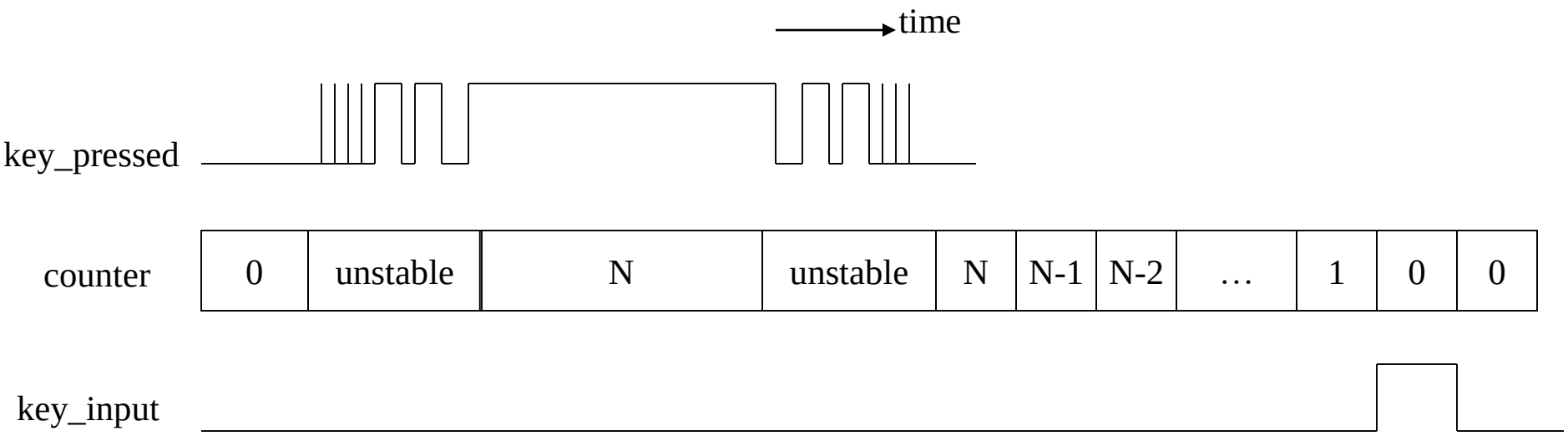
- the de-bounce filter

How to filter-out multiple pulses from a press



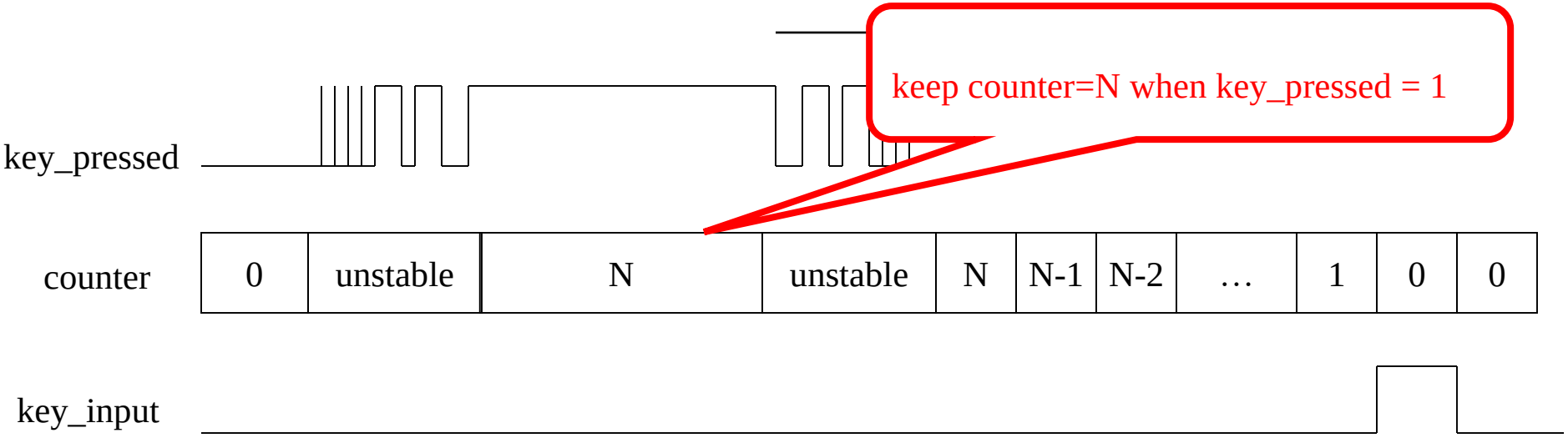
How to filter-out multiple pulses from a press

- use a counter to count the time to stable



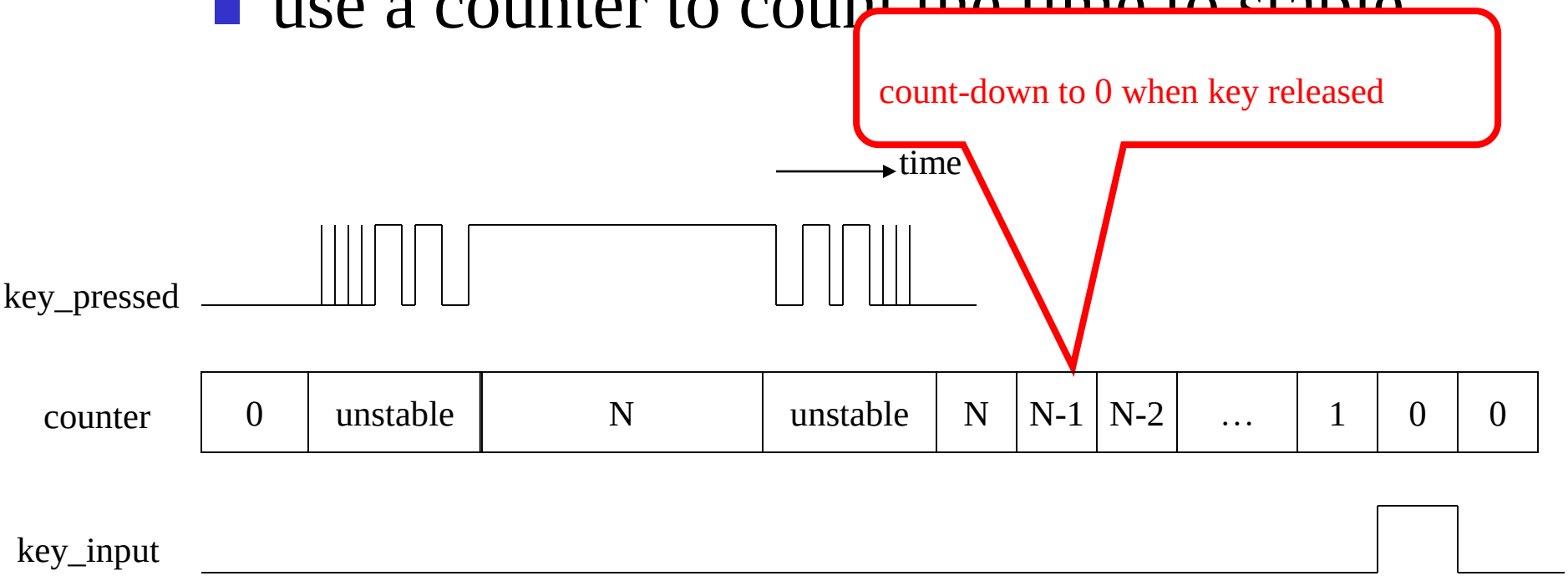
How to filter-out multiple pulses from a press

- use a counter to count the time to stable



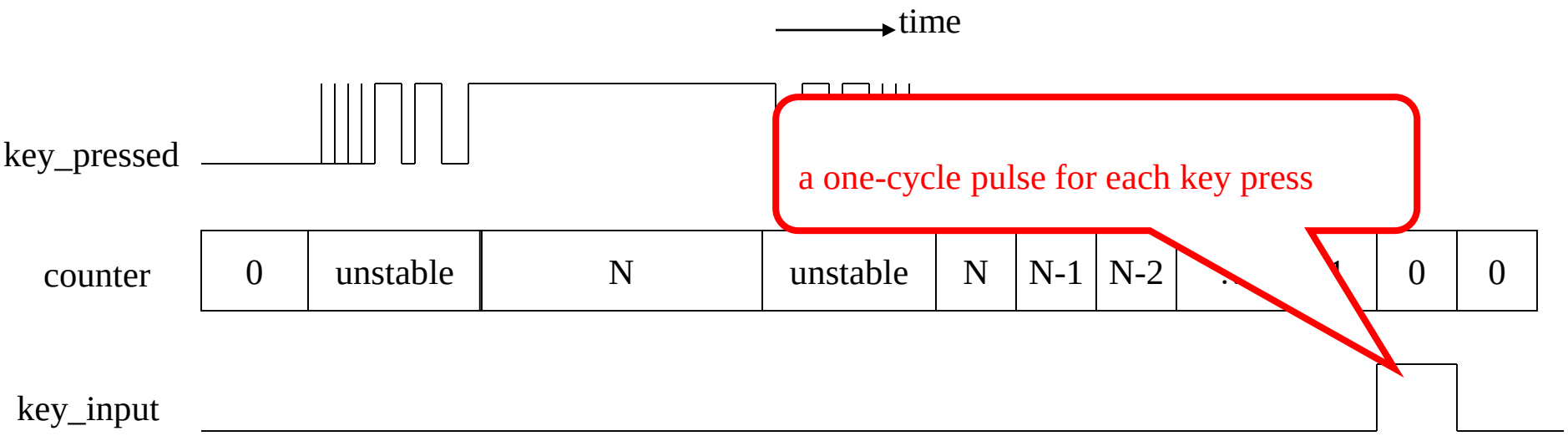
How to filter-out multiple pulses from a press

- 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

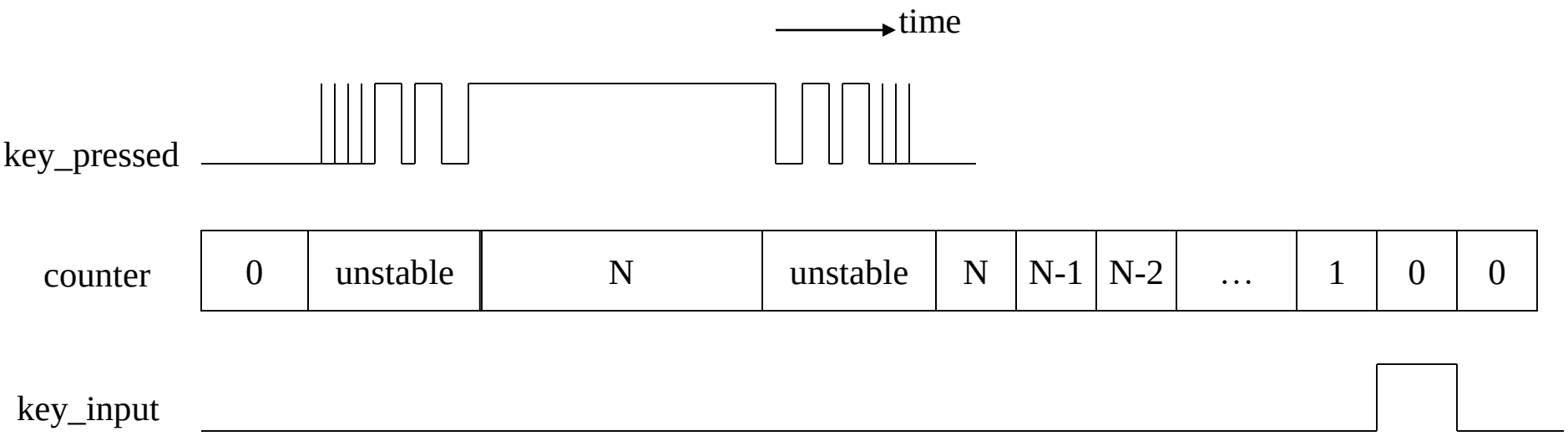


De-Bounce Filter

How to program

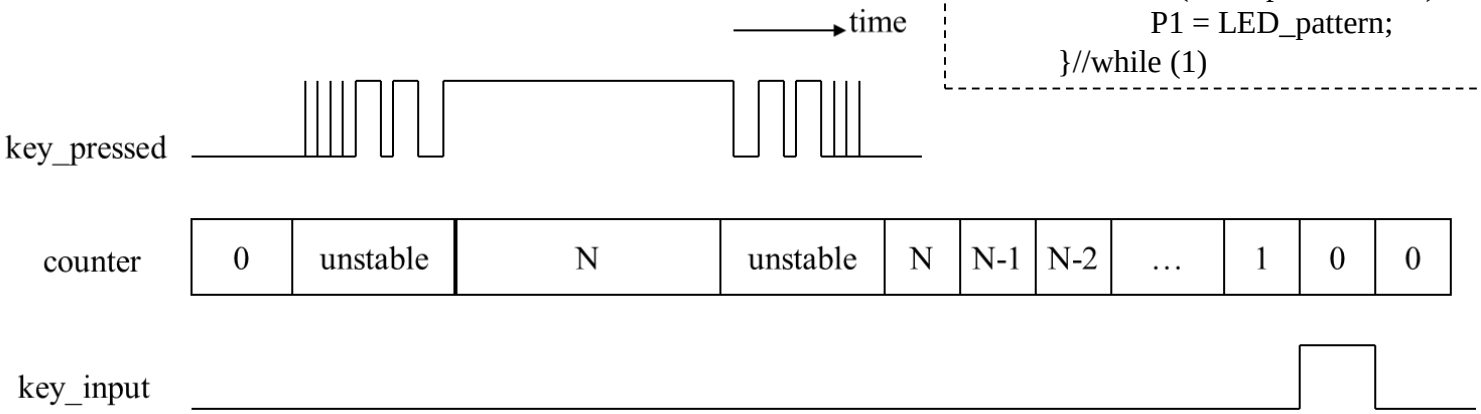
How to filter-out multiple pulses from a press

- write a program for the hardware concept



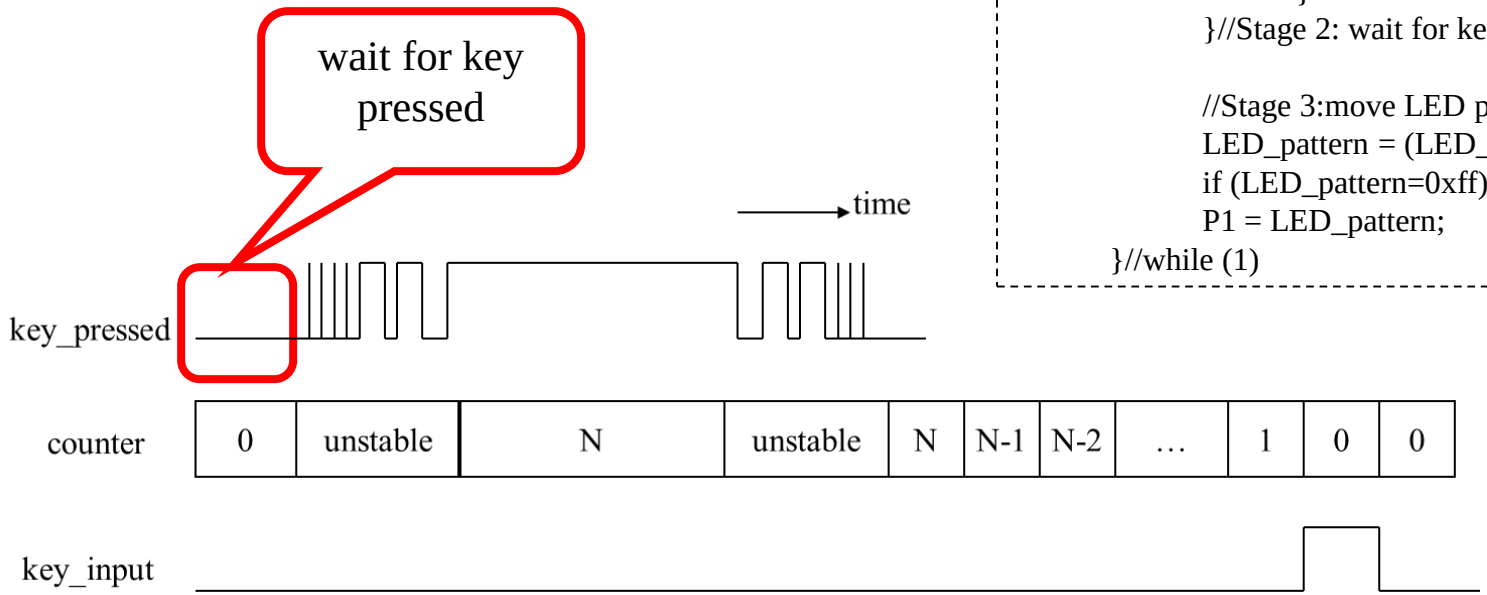
De-Bounce filter programming

```
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

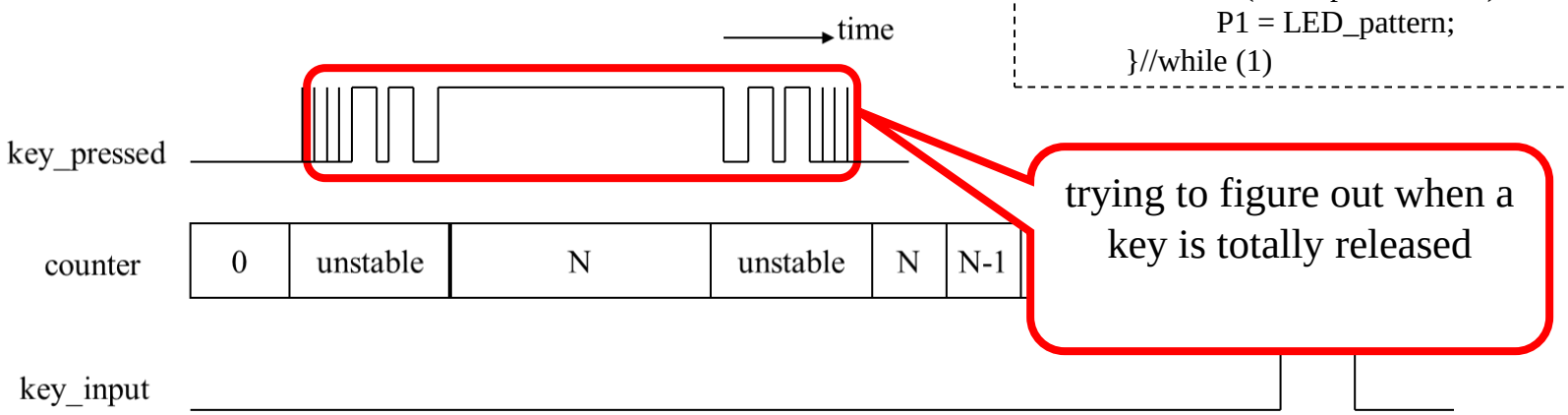
```
while (1) {
        //Stage 1: wait for a button pressed
        do {
                key_hold = P0;
        } while (!key_hold);

        //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

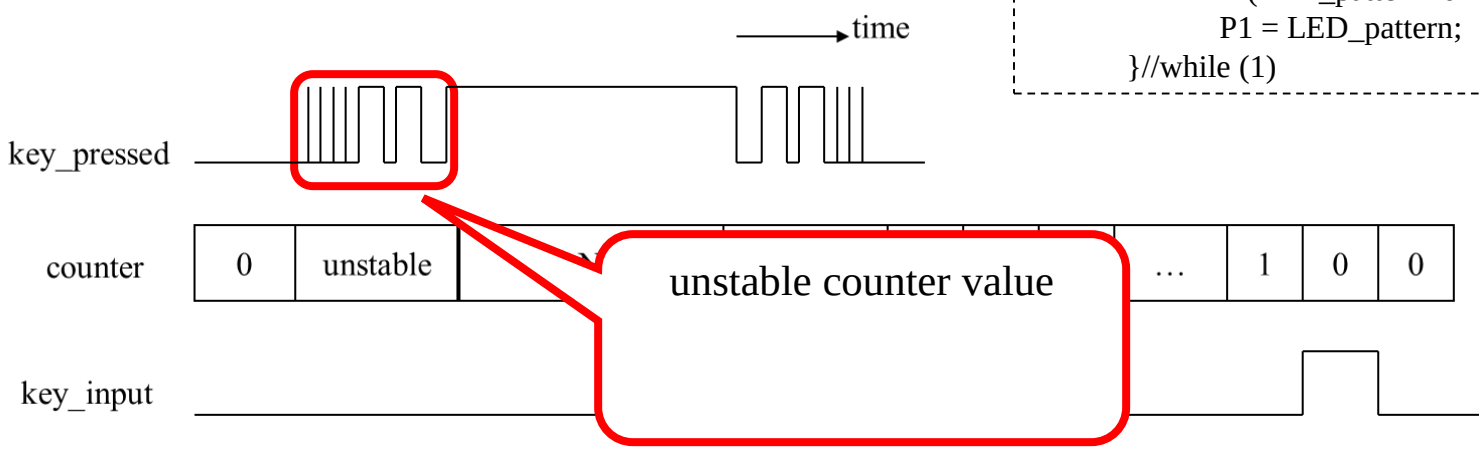
```
while (1) {
        //Stage 1: wait for a button pressed
        do {
                key_hold = P0;
        } while (!key_hold);

        //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)
```

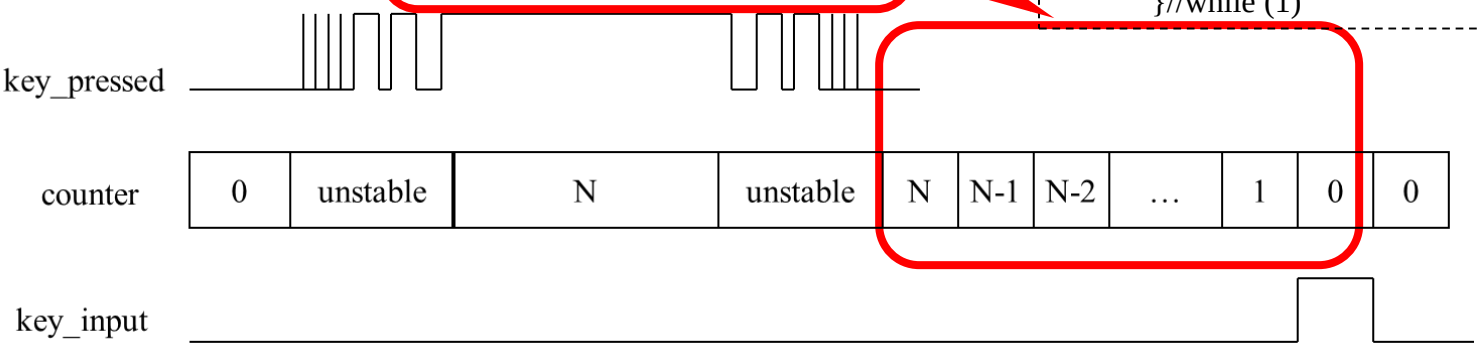


De-Bounce filter programming

- 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]);
}
```

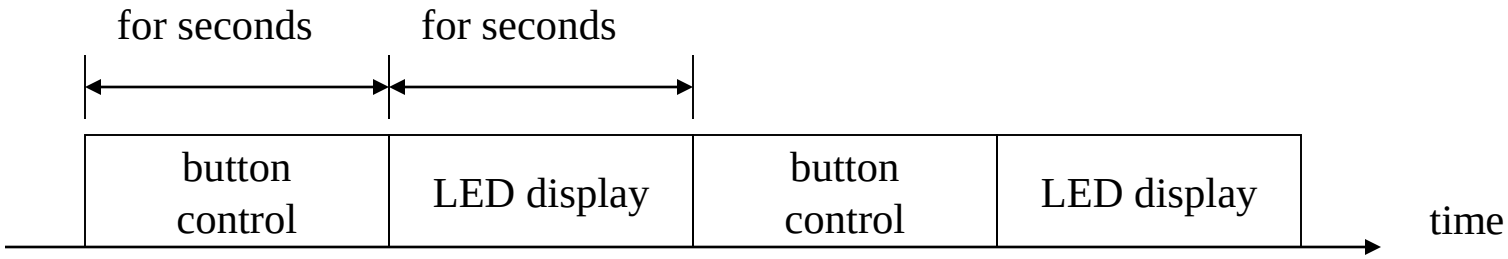
```
do {
        key_hold = ~P0;
} while (!key_hold);

//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 ();
}
```

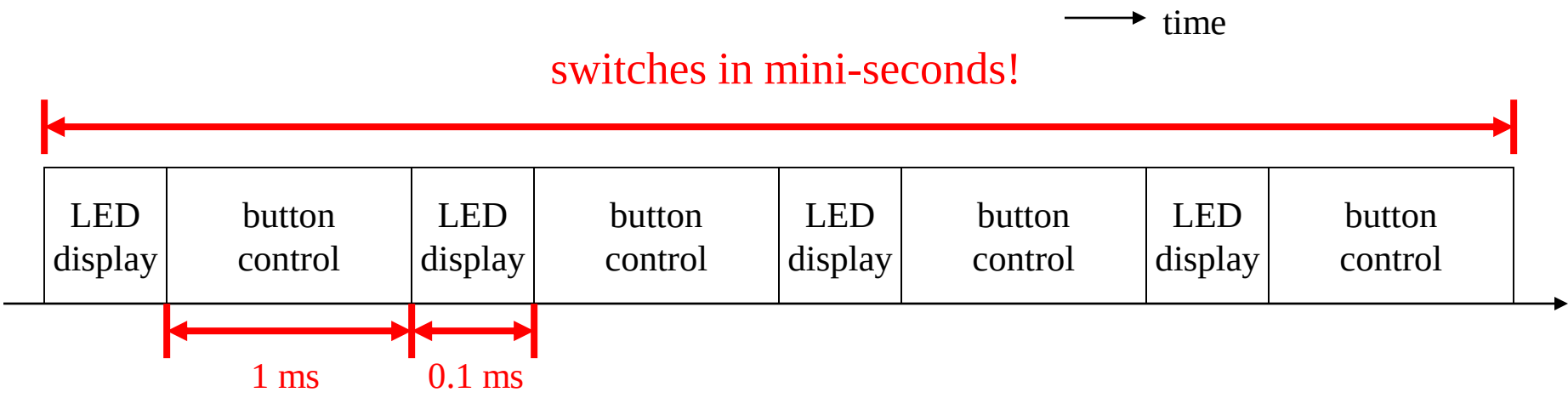
- You will never see button control and digit display work together



Time-sharing to control multiple I/O devices

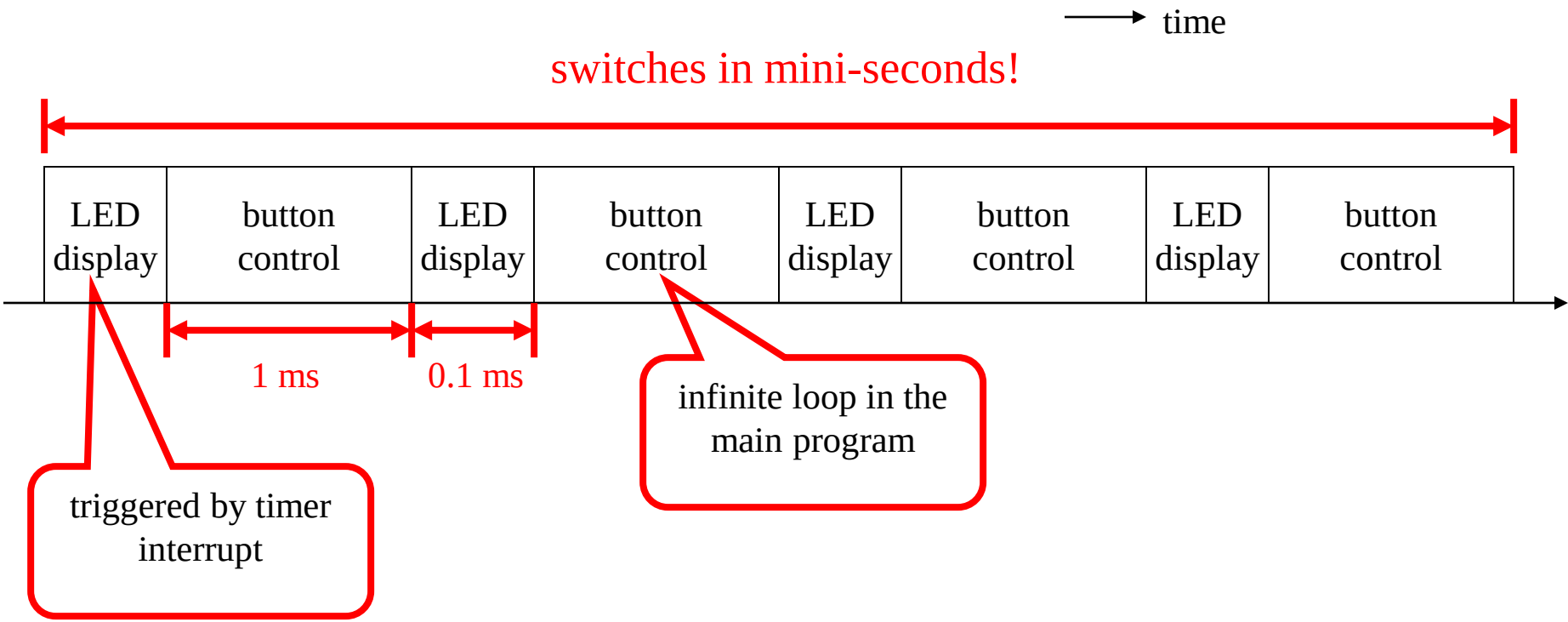
The correct scheme

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



A scheme for time sharing control

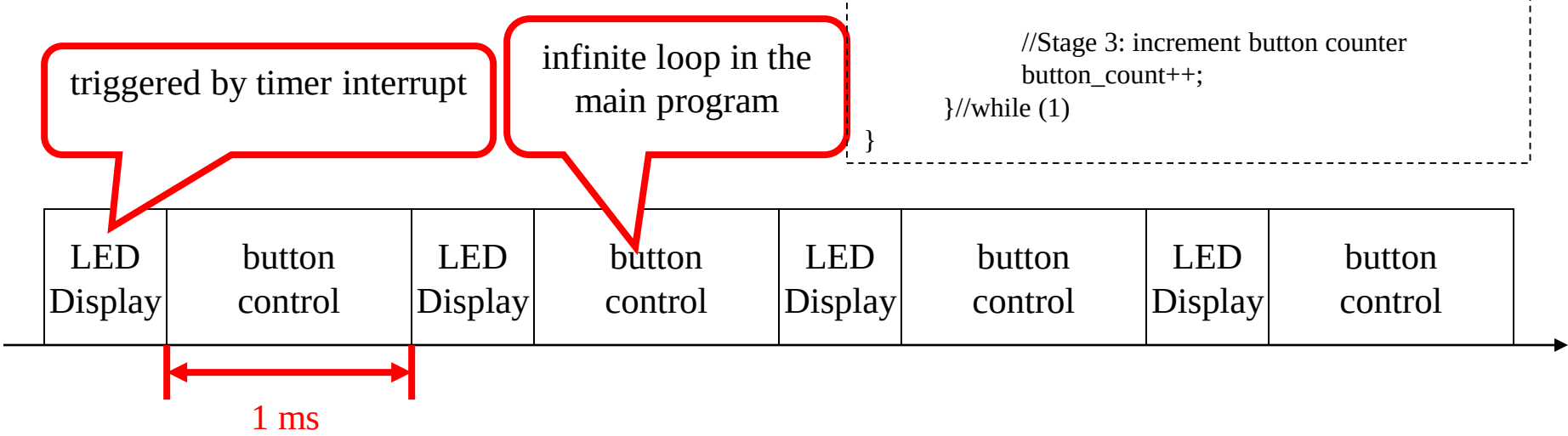
- main program: for button control
- timer ISR: to scan for digit display



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)
}
```



Appendix: Programming 8051 in C Language

Programming 8051 in C : Access Control Registers

- You have to #include "C8051F040.h
- In memory-mapped I/O
- Just access through a pointer

```
char *p;

p = (char*) 0x80;
*p = 0xaa;
```

Programming 8051 in C : Setup Interrupt Service Routines

- use “interrupt” directive
- The ISR table of Keil-C compiler: http://www.keil.com/support/man/docs/c51/c51_le_interruptfuncs.htm
- Check demo LED_shift

```
void
Timer0_ISR () interrupt 1
{
    count++;

    if (count==4) {
        count = 0;
        status = status>>1;
        if (status==0) status=0x80;
    }
```

Lab Requirements

- Basic Part: (80%)
  - 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, …
- Advanced Part: (20%)
  - The LED runs automatically
  - Hint: use timer interrupt to make two I/O devices work simultaneously

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: 2021/12/8 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: 2021/12/15 23:00 (不收遲交)
- Upload to e-learning system