**Bounce State Machine**

Program Enlisted Figures

- Asynchronous - any input can occur at any time.
- Synchronous - inputs and outputs
- Sequence of inputs and outputs are: UNKNOWN, UNREACHABLE, UNRESPONDABLE.
- **No "end" or "exit"**

Inputs: mouse/trackpad, will be; light sensor, etc.
Other inputs: short, button, touchscreen,疼痛

Outputs: update display, turn motor on/off

Mouse family, in good shape, ready to physically
Euler at Fermat Point

Conceptual Framework — Local methodology

Exact way to improve and doing programs (not machine)

Key point: Exploring "proper" data

Make it clear and low-level (find and square)

Make debugging easier

RULE #4

Tasks break down into two kinds of
clauses:

1. End Decision? Fast
2. Elsewhere

while (task not finished) {
    
    
    
...
Corollary to Rule III

- Keep sunk deadline at service rates as short as possible.
- Make the NWRB UNIQUE.

RTC - Run to completion
atomic code

Code Code Regime Structure

initialize all hardware and software
which (0) /

Fail if events
Service ends: STREAMING (EVENTS)
State Machine

- Description of an abstract machine
- At any point in time, it can be in one and only one of a fixed number of states.
- Next state in a progression depends only on the current state and the input (events)
- Directed transitions describe how one state to next
Static enum \{ none, right, one right, two right, open, left, end\} switch

\{ none, right \}
void main()
{
    Init all hardware and software;
    while(1)
    {
        wait = checkInput();
        runRobotStateMachine(wait);
    }
}