2020/2021 Joana Mesquita

Mouse (3)

Funções

→ int mouse_test_packet (uint32_t cnt)

✗ int mouse_test_remote (uint16_t period, uint8_t cnt)

→ int mouse_test_async (uint8_t idle_time)

→ int mouse_test_gesture (uint8_t x_len, uint8_t tolerance)

- O rato tem o seu próprio chip:
  - → guarda os estados dos botões
  - → guarda os movimentos na placa (em x e em y)

→ Esta informação é enviada para o PC através de packets de 3 bytes

Packet: (PS/2)

	7	6	5	4	3	2	1	0
B1	Yov	Xov	YSB	XSB	1	MB	RB	LB
B2	Δx (em complemento para 2)							
B3	Δy (em complemento para 2)							

O rato pode operar em:

→ Stream mode (default) - O rato envia data packets de maneira constante sempre que há um "mouse event" (i.e. mudança do estado dos botões ou mudança de posição rel. do rato)

→ Remote mode - O rato envia packets quando estes são pedidos

- Em ambos os casos, cada byte do data packet vai ser colocado no output buffer (mesmo q/ Keyboard)

arc

• O rato usa IRQ12, Atenção!! Até agora temos usado a IRQ como bit-mapa para os interrupts, mas como o bit-mapa tem de ser um nº entre 0 e 7, não podemos usar a IRQ12 como bit-mapa!!

Enabling and Desabling the mouse

• Para receber interrupts do mouse é necessário ativar o mouse data reporting.

- O mouse possui o mesmo argument buffer e command buffer bem como status register e output buffer que o keyboard.

Como escrever para o mouse?

→ Para escrevermos para o mouse e não para a keyboard é necessário enviar o comando 0xD4 para o command buffer e depois o enable data reporting para o argument buffer.

• Se quisermos escrever outro argumento temos de escrever denovo o 0xD4

→ O controlador depois de receber o argumento envia no output buffer um acknowledgment, se este não indica que está tudo okay deve-se repetir o processo de escrever o argumento.

Nota: Escrever estes argumentos e comando implica ler o status register e verificar se o input buffer está vazio.

Ler do output buffer implica verificar se o output buffer está cheio e se não há erro de paridade ou timeout.

Argumentos do comando 0xD4: Computer Labs: The PS/2 Mouse (OSDev)

→ Destiveas ? → é necessário desativar data reporting e colocar o rato em streaming mode.

int (mouse_test_packet)(uint32_t cnt)

função → organiza packets do rato num estrutura apropriada e dá print destes.

1º → ativar o data reporting do mouse

2º → subscrever interrupts do rato e colocar como condição para terminar o ciclo que cnt packets tenham sido lidas.

3º → ler os packets e organizá-los (tip: para passar de complemento para 2 negativo para um decimal negativo subtrair 256).

4º → imprimir packet

5º → terminar a subscrição do mouse.

6º → desativar o data reporting do mouse.

int (mouse_test_async)(uint8_t idle_time)

função → mesma que mouse_test_packet mas o programa termina quando não se recebe um packet durante idle_time.

→ Igual ao mouse_test_packet mas também tem interrupts do timer.
→ O custo do tempo deve ser estado se um packet for chamado.

int (mouse_test_gesture)(uint8_t xe_len, uint8_t tolerance)

função → termina quando a rato realiza um certo número de instruções requisitadas.

(1º) → Ativa data reporting, subscrever, etc.

(2º) → O rato deve ~~não~~ seguir instruções nesta ordem

(1º) - prime o botão esquerdo

(2º) - move o rato na diagonal para cima ↗ (e e g devem ser sempre positivos c/ pequenas imprecisões ñ maiores que a tolerance (i.e. se a tolerance = 3 e g = -2 o movimento pode continuar).

• até o movimento em x ter sido ≥ x_len

(3º) - larga o botão esquerdo

(4º) - prime o botão direito

Nota: movimentos verticais entre o 3º e 4º pares não devem ser tidos em conta.

(5º) - move o rato na diagonal para baixo ↘ (e é positivo e g é negativo (c/ tolerância para ambos)

• até o movimento em x ter sido ≥ x_len

(6º) - larga o botão direito

Atenção!! Para ler um packet são necessárias 3 interrupts do mouse, só se pode verificar o gesto quando o packet está completo

(3º) → desativa data reporting e termina subscrição.