Informe Técnico Nº 1

Dispositivos de Entrada de Información

Sofía Escudero, Nael Pighin, Martín Reus
Facultad de Ingeniería y Ciencias Hídricas

30 de noviembre de 2022

Resumen

En este informe se explicará, en primer lugar, el funcionamiento de una pantalla táctil capacitiva. En segundo lugar se hablará sobre como puede extenderse su principio de funcionamiento para utilizarlo en el reconocimiento de gestos. Por otro lado, en este informe también se escribirá un programa en Assembler de RISC-V que implemente el controlador de la matriz del teclado compacto Cherry G84-4400.

1. Sección de teoría

1.1. Pantallas táctiles

Para comenzar puede definirse que una **pantalla táctil** es un dispositivo que, a través del toque directo de su superficie, permite la entrada y salida de datos y órdenes al dispositivo, a la vez que muestra los resultados. Esto significa que actúa como un periférico de entrada y de salida simultáneamente. En la actualidad, las pantallas táctiles son utilizadas en una amplia variedad de dispositivos de uso común como teléfonos celulares, tablets, computadoras personales y lectores electrónicos, entre otros.

Una de las tecnologías más utilizadas a la hora de implementar pantallas táctiles es la capacitiva. Las **pantallas capacitivas** consisten en una capa de aislante, que puede ser de cristal, recubierto con múltiples electrodos hechos de un material transparente. La configuración más sencilla de la pantalla consiste en dos matrices de electrodos colocados a 90^{o} entre sí. Este sistema aprovecha la capacidad conductora del cuerpo humano, ya que al tocar la superficie del conductor se produce una distorsión en el campo electrostático de la pantalla. La distorsión producida se mide por el cambio de capacitancia y esto permite que pueda determinarse en que posición de la pantalla fue hecho el toque.

Existen diferentes tecnologías para determinar la posición en que se produjo el toque. Una de ellas es la *capacitancia superficial*, para la cual solo un lado del aislante está recubierto con material conductor, y se aplica un pequeño voltaje a esta capa, lo que resulta en un campo electrostático uniforme. El controlador del sensor puede determinar la ubicación del toque indirectamente a partir del cambio en la capacitancia medido desde las cuatro esquinas del panel: cuanto mayor sea el cambio en la capacitancia más cerca estará el toque de esa esquina. Es moderadamente duradero, pero tiene una resolución baja y necesita calibración.

Puede también utilizarse la tecnología de *autocapacitancia*, donde las columnas y filas funcionan de manera independiente. La corriente detecta la carga capacitiva de un dedo en cada columna y fila. Esto genera la desventaja de ser incapaz de resolver con precisión toques utilizando más de un dedo simultáneamente, lo que resulta en un "efecto fantasma" o en una detección de ubicación mal colocada

Otra de las tecnologías utilizadas es la *capacitancia proyectada* en la cual se utilizan dos capas paralelas separadas de material conductor con pistas perpendiculares para formar la cuadrícula. Esto permite pueda operarse con la pantalla sin hacer contacto directo, de modo que las capas conductoras pueden recubrirse con capas aislantes protectoras, operando bajo protectores de pantalla. Sin embargo, las manchas conductoras pueden interferir con el rendimiento.

La última de las tecnologías que se nombrarán es la *capacitancia mutua*, que posee la ventaja de permitir la operación multitáctil. Aquí se cuenta con un condensador en cada intersección de cada fila y cada columna. Se aplica un voltaje a las filas o columnas. Llevar un dedo o un lápiz conductor cerca de la superficie del sensor cambia el campo eléctrico local, lo que reduce la capacidad mutua. El cambio de capacitancia en cada punto individual de la red se puede medir para determinar con precisión la ubicación del contacto al medir el voltaje en el otro eje.

1.2. Reconocimiento de gestos

El reconocimiento de gestos consiste en interpretar los gestos humanos a través de algoritmos matemáticos. Los gestos son movimientos del cuerpo humano que contienen información y pueden ser movimientos corporales, emociones faciales, gestos de la mano, entre otros. Presionar una tecla en un teclado no se considera un gesto porque el movimiento en un dedo no se observa ni es significativo: lo único que importa es qué tecla se presionó y no cómo o con qué intencionalidad.

El reconocimiento de gestos se puede ver como una forma en que las computadoras comienzan a entender el lenguaje del cuerpo humano, creando así un puente para la interacción entre máquinas y humanos mucho más natural que otras interfaces comúnmente limitadas al ratón (mouse) o al teclado.

Las áreas que más utilizan el reconocimiento de gestos son el sector automotriz, el sector de tránsito, el sector de videojuegos (sobre todo las empresas fabricantes de consolas y controles para jugar), las empresas de celulares con el desbloqueo facial. Incluso se utiliza para la interpretación en lenguaje de señas, agregando un elemento de inclusividad a esta tecnología.

Los seguimientos de los gestos pueden hacerse mediante distintas herramientas o tecnologías de sensado:

1.2.1. Guantes

Proporcionan información sobre la posición y movimiento de las manos con dispositivos de seguimiento inercial. Dependiendo de la precisión que se necesite o qué tipo de gesto de las manos se deba capturar, el guante puede

llegar a captar movimientos de las falanges con gran precisión o ser más simple. También pueden generar la sensación de tacto a la persona.

1.2.2. Cámaras de profundidad

Mediante estas cámaras se generan mapas de profundidad de lo que se ve a través de la cámara a corta distancia, los cuales luego se usan para generar una representación 3D de lo que se está viendo. Funciona de la siguiente manera. el sensor de la cámara emite una señal luminosa, que golpea al sujeto y vuelve al sensor. A continuación, se mide el tiempo que tarda en rebotar y así se obtiene el mapa de profundidad. Un ejemplo popularizado es la cámara de detección de movimiento de la consola PlayStation.

1.2.3. Sistemas guiados a mano

Actúan como una extensión del cuerpo: la persona mueve el controlador y sus gestos son capturados por el sensor de aceleración como sucede con los mandos de las consolas (por ejemplo, el Wiimote de la consola Wii).

1.2.4. Pantallas táctiles

En los teléfonos y tablets se utilizan las pantallas táctiles, que se pueden usar con "gestos táctiles" (patrones de toques captados por los sensores y que la aplicación interpreta como un gesto específico). Las pantallas multitáctiles ofrecen la detección de varias huellas dactilares independientes simultáneamente, lo cual se puede usar, por ejemplo, para ampliar o minimizar un objeto en pantalla (por ejemplo, hacerle zoom a una foto o minimizar una ventana).

Últimamente se han implementado muchas modificaciones a los celulares con el fin de disminuir el uso de la pantalla táctil. Por ejemplo, según el modelo, podemos prender la linterna del celular agitándolo de izquierda a derecha varias veces en vez de ir a buscarla en el menú. Otro ejemplo es que la pantalla se prenda sin tener que tocarla, sólo con la acción de levantar el dispositivo, o poder desbloquearlo sin tener que poner el código de seguridad sino con la detección facial que brinda la cámara.

2. Sección de práctica

2.1. Consigna

Utilizando un procesador RISC-V se escribió un programa en Assembler que implementa el controlador de la matriz del teclado compacto Cherry G84-4400, tal que:

- Los bits 0 a 3 del puerto paralelo ubicado en la posición de memoria 0x10000 se utilizan para controlar un decodificador 4 a 16 para activar las columnas de la matriz del teclado;
- Los bits 4 a 7 del puerto paralelo ubicado en la posición de memoria 0x10000 se utilizan para controlar el encendido de los led indicadores. Un estado lógico alto (1) en cualquiera de estas líneas enciende el led correspondiente, mientras que un estado lógico bajo (0) lo apaga;
- Los bits 0 a 5 del puerto paralelo ubicado en la posición de memoria 0x10001 se utilizan para leer las filas de la matriz del teclado, una fila por cada bit. Una tecla está apretada cuando un lee un estado lógico alto en la fila correspondiente;
- Los bits 6 a 7 del puerto paralelo ubicado en la posición de memoria 0x10001 se utilizan para leer el estado de los botones del trackball. En la línea se lee un estado lógico alto cuando el botón asociado ha sido pulsado, en caso contrario se lee un estado lógico bajo;
- Los puertos paralelos ubicados en posiciones de memoria 0x10002 y 0x10003 contienen la información del desplazamiento angular del trackball en los ejes X e Y.

2.2. Implementación

```
.data
.data

## Mensajes de salida (los mensajes de las teclas tienen
    todos el mismo tamaño, 36 Bytes)
```

```
# Primera Fila
pEsc:       .asciz "Se presionó la tecla Escape      \n"
                .align 2
pF1:        .asciz "Se presionó la tecla F1          \n"
                .align 2
pF2:        .asciz "Se presionó la tecla F2          \n"
                .align 2
pF3:        .asciz "Se presionó la tecla F3          \n"
                .align 2
pF4:        .asciz "Se presionó la tecla F4          \n"
                .align 2
pF5:        .asciz "Se presionó la tecla F5          \n"
                .align 2
pF6:        .asciz "Se presionó la tecla F6          \n"
                .align 2
pF7:        .asciz "Se presionó la tecla F7          \n"
                .align 2
pF8:        .asciz "Se presionó la tecla F8          \n"
                .align 2
pF9:        .asciz "Se presionó la tecla F9          \n"
                .align 2
pF10:       .asciz "Se presionó la tecla F10         \n"
                .align 2
pF11:       .asciz "Se presionó la tecla F11         \n"
                .align 2
pF12:       .asciz "Se presionó la tecla F12         \n"
                .align 2
pPrtScr:    .asciz "Se presionó la tecla PrtScr      \n"
                .align 2

# Segunda Fila
pPause:     .asciz "Se presionó la tecla Pause     \n"
                .align 2
pCircunflex:.asciz "Se presionó la tecla ^         \n"
                .align 2
p1:         .asciz "Se presionó la tecla 1         \n"
                .align 2
p2:         .asciz "Se presionó la tecla 2         \n"
                .align 2
p3:         .asciz "Se presionó la tecla 3         \n"
                .align 2
p4:         .asciz "Se presionó la tecla 4         \n"
                .align 2
p5:         .asciz "Se presionó la tecla 5         \n"
                .align 2
p6:         .asciz "Se presionó la tecla 6         \n"
                .align 2
p7:         .asciz "Se presionó la tecla 7         \n"
                .align 2
```

```
p8:          .asciz "Se presionó la tecla 8         \n"
                .align 2
p9:          .asciz "Se presionó la tecla 9         \n"
                .align 2
p0:          .asciz "Se presionó la tecla 0         \n"
                .align 2
pEszett:     .asciz "Se presionó la tecla ß         \n"
                .align 2
pTilde:      .asciz "Se presionó la tecla "         \n"
                .align 2
pBackspace:  .asciz "Se presionó la tecla Backspace \n"
                .align 2
pHome:       .asciz "Se presionó la tecla Home      \n"
                .align 2

# Tercera Fila
pTab:        .asciz "Se presionó la tecla Tab       \n"
                .align 2
pQ:          .asciz "Se presionó la tecla q         \n"
                .align 2
pW:          .asciz "Se presionó la tecla w         \n"
                .align 2
pE:          .asciz "Se presionó la tecla e         \n"
                .align 2
pR:          .asciz "Se presionó la tecla r         \n"
                .align 2
pT:          .asciz "Se presionó la tecla t         \n"
                .align 2
pZ:          .asciz "Se presionó la tecla z         \n"
                .align 2
pU:          .asciz "Se presionó la tecla u         \n"
                .align 2
pI:          .asciz "Se presionó la tecla i         \n"
                .align 2
pO:          .asciz "Se presionó la tecla o         \n"
                .align 2
pP:          .asciz "Se presionó la tecla p         \n"
                .align 2
pUDieresis:  .asciz "Se presionó la tecla ü         \n"
                .align 2
pMas:        .asciz "Se presionó la tecla +         \n"
                .align 2
pEnter1:     .asciz "Se presionó la tecla Enter     \n"
                .align 2
pPgUp:       .asciz "Se presionó la tecla PgUp      \n"
                .align 2

# Cuarta Fila
pCaps:       .asciz "Se presionó la tecla Caps      \n"
```

```
                .align 2
pA:         .asciz "Se presionó la tecla p         \n"
                .align 2
pS:         .asciz "Se presionó la tecla s         \n"
                .align 2
pD:         .asciz "Se presionó la tecla d         \n"
                .align 2
pF:         .asciz "Se presionó la tecla f         \n"
                .align 2
pG:         .asciz "Se presionó la tecla g         \n"
                .align 2
pH:         .asciz "Se presionó la tecla h         \n"
                .align 2
pJ:         .asciz "Se presionó la tecla j         \n"
                .align 2
pK:         .asciz "Se presionó la tecla k         \n"
                .align 2
pL:         .asciz "Se presionó la tecla l         \n"
                .align 2
pODieresis: .asciz "Se presionó la tecla ö         \n"
                .align 2
pADieresis: .asciz "Se presionó la tecla ä         \n"
                .align 2
pNumeral:   .asciz "Se presionó la tecla #         \n"
                .align 2
pEnter2:    .asciz "Se presionó la tecla Enter     \n"
                .align 2
pPgDn:      .asciz "Se presionó la tecla PgDn      \n"
                .align 2

# Quinta Fila
pShift:     .asciz "Se presionó la tecla Shift     \n"
                .align 2
pMenor:     .asciz "Se presionó la tecla <         \n"
                .align 2
pY:         .asciz "Se presionó la tecla y         \n"
                .align 2
pX:         .asciz "Se presionó la tecla x         \n"
                .align 2
pC:         .asciz "Se presionó la tecla c         \n"
                .align 2
pV:         .asciz "Se presionó la tecla v         \n"
                .align 2
pB:         .asciz "Se presionó la tecla b         \n"
                .align 2
pN:         .asciz "Se presionó la tecla n         \n"
                .align 2
pM:         .asciz "Se presionó la tecla m         \n"
                .align 2
```

```
pComa:      .asciz "Se presionó la tecla ,         \n"
                .align 2
pPunto:     .asciz "Se presionó la tecla .         \n"
                .align 2
pGuion:     .asciz "Se presionó la tecla -        \n"
                .align 2
pShiftDer:  .asciz "Se presionó la tecla ShiftDer \n"
                .align 2
pUp:        .asciz "Se presionó la tecla up       \n"
                .align 2
pEnd:       .asciz "Se presionó la tecla End      \n"
                .align 2

# Sexta Fila
pFn:        .asciz "Se presionó la tecla Function \n"
                .align 2
pCtrl:      .asciz "Se presionó la tecla Control  \n"
                .align 2
pAlt:       .asciz "Se presionó la tecla Alt      \n"
                .align 2
pSpace1:    .asciz "Se presionó la tecla Espacio  \n"
                .align 2
pSpace2:    .asciz "Se presionó la tecla Espacio  \n"
                .align 2
pSpace3:    .asciz "Se presionó la tecla Espacio  \n"
                .align 2
pSpac4:     .asciz "Se presionó la tecla Espacio  \n"
                .align 2
pSpace5:    .asciz "Se presionó la tecla Espacio  \n"
                .align 2
pSpace6:    .asciz "Se presionó la tecla Espacio  \n"
                .align 2
pAltGr:     .asciz "Se presionó la tecla Alt Gr   \n"
                .align 2
pInsert:    .asciz "Se presionó la tecla Insert   \n"
                .align 2
pDelete:    .asciz "Se presionó la tecla Delete   \n"
                .align 2
pLeft:      .asciz "Se presionó la tecla Left     \n"
                .align 2
pDown:      .asciz "Se presionó la tecla Down     \n"
                .align 2
pRight:     .asciz "Se presionó la tecla Right    \n"
                .align 2


aLedNum:    .asciz "LED Num activado   \n"
                .align 2
aLedCaps:   .asciz "LED Caps activado  \n"
```

```
                .align 2
aLedScroll: .asciz "LED Scroll activado\n"
                .align 2
aLedPad:    .asciz "LED Pad activado   \n"
                .align 2

aBtnDer:    .asciz "Activado el Botón Derecho del
   Trackball  \n"
                .align 2
aBtnIzq:    .asciz "Activado el Botón Izquierdo del
   Trackball\n"
                .align 2


desplazamientoX: .asciz "El Trackball se desplazó en X: "
                    .align 2
desplazamientoY: .asciz "\nEl Trackball se desplazó en Y:
    "
                    .align 2


## Se carga en el registro t0(x5) los datos necesarios
#  para activar la tecla 'm', con los LEDS Num y Pad
#  activados.
#  También estará el Botón derecho del Trackball activado
#  y habrá un cierto desplazamiento del Trackball.

#   Para la tecla 'm'
#       Novena col  -> col = 1000 = 8
#       Quinta fila -> fila = 010000 = 32
#
#   Leds que se activan
#       leds = Num|Caps|Scroll|Pad -> leds = 1001 = 9
#
#   Botones que se activan en el Trackball
#       btnTrack = BtnIzq|BtnDer -> btnTrack = 01 = 1
#
#   Desplazamiento en X e Y del Trackball
#       X =  103 -> 0110 0111 -> 0x67
#       Y = -95 -> 1010 0001 -> 0xA1


# Los valores se guardaran en  direcciones de memoria
# simulando el uso de los puertos correspondientes.
# No son las mismas direcciones que se piden en la
# consigna porque el RARS no permite utilizar las
# direcciones de memoria 0x00010000 a 0x00010003 por
# lo que se usaran las direcciones 0x10000000 a
# 0x10000003, pero en un caso real podrían usarse las
```

```
# de la consigna de la misma forma que las utilizadas en
# este trabajo.

.extern LedCol, 1   # Byte formado por (leds y col)
.extern BtnFila, 1  # Byte formado por (btnTrack y fila)
.extern posX, 1     # Byte formado por (X)
.extern posY, 1     # Byte formado por (Y)


.text
    li a7,4 # Para las llamadas ecall del RARS
    addi a1, zero, 36 # Tamaño mensaje al pulsar teclas

#   Se guardan los valores del registro t6 en la
#   dirección de los puertos, para simular una entrada
#   del teclado.

#   t6 = 0 x (Y | X | btnTrack y fila | leds y col)
#      = 0 x (a1 | 67 | 50 | 98)
    li t6, 0xA1675098
    sw t6, LedCol, t2


## Comienza el proceso de lectura de puertos
#   Se cargan los valores desde la dirección de los
#   puertos
    lb t0, LedCol
    lb t1, BtnFila
    lb t2, posX
    lb t3, posY


# Lectura del teclado

    addi s0, x0, 15 # Cantidad final columnas (16-1 = 15
       porque empieza desde 0 a contar)

    addi s1, x0, 64 # Valor final de filas (2^6, porque
       corresponde a un numero de 6 bits)

    addi s2, x0, 0  # Inicializando contador de columnas

    addi s3, x0, 0  # Inicializando contador de filas

    addi t6, x0, 1  # Inicializando el valor del bit
       activado correspondiente a la fila (empieza en
       000001)

    andi s4, t0, 0xF # Cargar en el reg s4 (x20) el valor
```

```
        de la columna de la tecla (AND entre lo que hay
       en t0 y 1111)

    andi s5, t1, 0x3F # Cargar en el reg s5 (x21) el
       valor de la fila de la tecla (AND entre lo que
       hay en t1 y 00111111)

loopFila:
    bne t6, s5, aumentarFila # Si la fila actual no es la
        fila activada entonces pasar a la que sigue.

loopCol:

    bne s2, s4, aumentarCol # Si la col actual no es la
       col activada entonces pasar a la que sigue.

    # Si se logra llegar aquí es porque se encontró la
       fila y columna de la tecla pulsada.
    # Se procede a imprimir un mensaje correspondiente a
       la tecla.
    la a0, pEsc

    mul t5, s3, s0
    add t5, t5, s2
    mul t5, t5, a1
    add a0, a0, t5

    jal x0, finLoopFila # Termina el loop que recorre el
       teclado
aumentarCol:

    addi s2, s2, 1 # Sumar 1 al numero de columna actual
    bne s2, s0, loopCol

aumentarFila:

    addi s3, s3, 1  # Sumar 1 al numero de fila actual
    slli t6, t6, 1  # Mover la fila actual una posicion a
        la izquierda para pasar a la siguiente
    bne t6, s1, loopFila

finLoopFila:
    ecall # Llama al mensaje de la tecla que fue
       encontrada


# Lectura del estado de los LEDS

    andi s0, t0, 0xF0   # Cargar en s0 los estados de los
```

```
        4 leds (4 bits)
    srli s0, s0, 4      # Reacomodar los bits para que
       queden al principio de la palabra
    addi s1, zero, 1    # Contador para recorrer los leds
    addi s2, zero, 16   # Valor con el cual comparar si
       el LED está encendido, y también la condicion de
       corte del loop
    la a0, aLedNum      # Dirección donde se encuentra el
        primer mensaje de "LED activado"
    addi a1, zero, 24   # Tamaño de los mensajes de "LED
       activado" (6 Bytes)

loopLeds:

    and s3, s0, s1      # Mascara para ver el estado solo
        del LED que toca en este ciclo del loop

    bne s3, s1, siguienteLed # Si no está activado se
       pasa al siguiente
    ecall           # Si lo está, se imprime el mensaje
       de que está encendido

siguienteLed:
    add a0, a0, a1      # Se pasa a la dirección del
       siguiente mensaje por si debe imprimirse
    slli s1, s1, 1      # Se desplaza hacia la derecha en
        1 bit el contador para pasar al siguiente LED
    beq s1, s2 finLoopLeds
    jal x0, loopLeds

finLoopLeds:


# Lectura del estado de los botones del Trackball

    andi s0, t1, 0xC0   # Cargar en s0 los estados de los
        2 botones
    srli s0, s0, 6      # Reacomodar los bits para que
       queden al principio de la palabra
    addi s1, zero, 1    # Contador para recorrer los
       estados de los botones
    addi s2, zero, 4    # Valor con el cual comparar si
       el botón está activado, y también la condicion de
        corte del loop
    la a0, aBtnDer      # Dirección donde se encuentra el
        primer mensaje de activación de los botones
    addi a1, zero, 44   # Tamaño de los mensajes de
       activación de los botones (11 Bytes)
loopBtn:
```

```
    and s3, s0, s1      # Máscara para ver el estado solo
        del botón que toca en este ciclo del loop

    bne s3, s1, siguienteBtn # Si no está activado se
       pasa al siguiente
    ecall           # Si lo está se imprime el mensaje de
        que está activado

siguienteBtn:
    add a0, a0, a1      # Se pasa a la dirección del
       siguiente mensaje por si debe imprimirse
    slli s1, s1, 1      # Se desplaza hacia la derecha en
        1 bit el contador para pasar al siguiente botón
    beq s1, s2 finLoopBtn
    jal x0, loopBtn

finLoopBtn:

# Lectura del desplazamiento del trackball

    la a0,desplazamientoX #Se carga la dirección del
       mensaje que corresponde al desplazamiento en X
        ecall # Se imprime el mensaje
        li a7,1 #Se cambia el tipo de dato a imprimir (
           pasa de ser String a entero)
    add a0,zero,t2 # Se guarda en a0 el valor del
       desplazamiento en X
    ecall # Se imprime el valor del desplazamiento en X
    li a7,4 #Se cambia el tipo de dato a imprimir (pasa
       de ser entero a String)
    la a0,desplazamientoY #Se carga la dirección del
       mensaje que corresponde al desplazamiento en X
    ecall # Se imprime el mensaje
    li a7,1 #Se cambia el tipo de dato a imprimir (pasa
       de ser String a entero)
    add a0,zero,t3 # Se guarda en a0 el valor del
       desplazamiento en Y
    ecall # Se imprime el valor del desplazamiento en Y
```