# Sistema de Alarma Domiciliaria
# Esquema Gral Hw y Sw

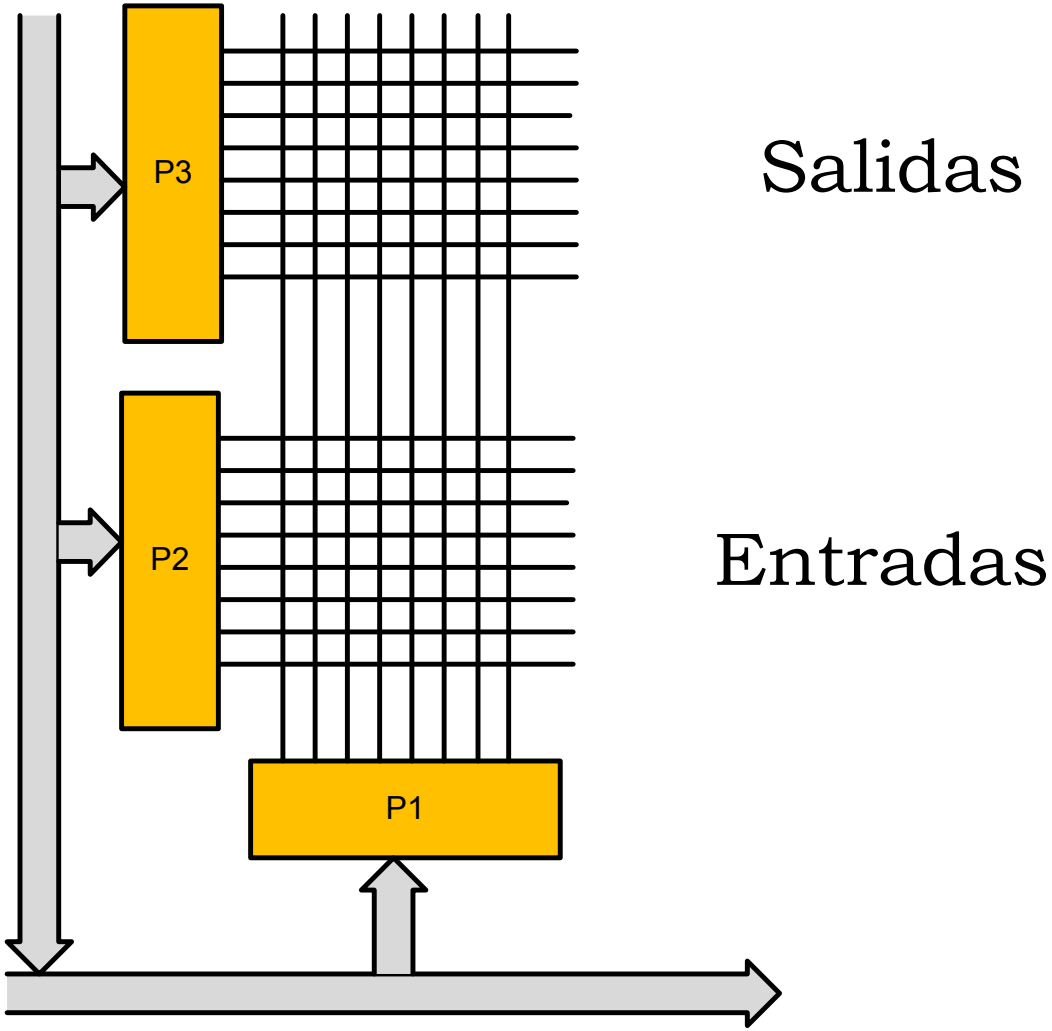
**Sistemas Embebidos Avanzados**

Eduardo Daniel Cohen
dcohen@herrera.unt.edu.ar
http://www.microprocesadores.unt.edu.ar/embebidosavanzados

# Temario

- Sistema de Múltiples E/S (Repaso)
  - Matriz de Entrada/Salida.
  - Teclado.
- Sw. de Base
  - Interrupciones y Rebotes.
  - Timers
  - Matriz Lógica de Estímulos.
  - Estructura de Datos

# Visión Global



# Ej. Disposición Física Entradas

| Posición Matriz | Número | Función |
|---|---|---|
| 0/0.0/3 | 0-3 | 1,4,7,F |
| 1/0,1/3 | 8-14 | 2,5,8,0 |
| 2/0.2/3 | 16-19 | 3,6,9,C |
| 3/0.3/3 | 24-27 | ///No Usar/// |
| 0/4.0/7 | 4-7 | P, F, E, X |
| 1/4, 1/5 | 12-14 | Sirena, Prueba, U-Estado, X |
| 2/4,2/7 | 20-23 | Fuego y Expansión F |
| 3/4/,3/7 | 28-31 | Fuego y Expansión F |
| 4/0 : 7/3 | 32-35, … 56,59 | Sensores Intrusión |
| 4/4 : 7/7 | 36,39 … 60,63 | Expansión Sensores |

**El teclado remoto debe conectarse como una matriz. de 4x4 (UTP).**
**4 contactos Fila 3 para 4 Leds**
**Se usan 6 teclas de función en un panel central**

**No usar. Se podrían usar, pero no hay UTP**
X: No usados
**¿Si en el futuro cambia el Hw de E/S?...**

**¿Cómo mantener el resto del sistema sin cambios?**

# Asignación Lógica de Input

| Número Lógico | Función |
|---|---|
| 0-9 | Teclas Numéricas |
| 10,15 | /// |
| 16 | Test |
| 17 | Sirena |
| 18 | Status |
| 19 | Status-A |
| 20 | Clear |
| 21 | Function |
| 22-23 | Armar-C, Forzar-C |
| 24 | Fuego Cocina |
| 25 | F. Living-Comedor |
| 26 | F. Dormitorios |
| 27-31 | Expansión Fuego |
| 32-48 | Sensores Intrusión |
| 50-63 | Expansión Sensores |

Teclas numéricas de 0-9

Filas para
- Números
- Funciones (ej TEST).
- Sensores.

Salidas lógicas:
Existe una correspondencia Biunívoca entre filas lógicas de sensores y filas de Leds que los representan.

# En temas 2 y 3

- Se aprende a obtener una matriz de transición de 0 a 1, MT.
  - Si toda tecla o sensor que se active pasa de 0 a 1.
  - Consituye un estímulo.
  - Además se cuenta con una matriz lógica de entrada, MEL.
- También se aprende a manejar
  - Estímulos percibidos.
  - Estado lógico percibido.
  - Son matrices lógicas.
    - Propias de la aplicación.

# Cola de Despacho.

- Se trata de una cola en que se sitúan estímulos y eventos terminados, pendientes de procesar por el sistema.
- Los estímulos se ingresarán en una cola de despacho, con su número de identificación y el número de tarea que los procesa.
  - Más una variable lógica=1 que indica que es un estímulo.
- ¿Cuál módulo de Sw los ingresa?

# ENTRADA

- Es el Software de Base.
  - Obtener ME, MT
  - Si MT=0 → nada para hacer, fin.
- Obtener MEL, MTL (con tabla conversión).
- A partir de la matriz de percepción, MP, obtiene
  - Matriz de Transición Lógica Percibida
    - MTLP = MTL x MP
  - La matriz de transición (estímulos) percibidos se obtiene desde MTL y no MEL.
    - Transición válida = estímulo
- Se incorpora así el concepto de Percepción.

# ENTRADA (II)

- Si MTLP=0 nada para hacer → fin.
- Recorre la Matriz de Estímulos percibidos (MTLP).
- Antes de encolarlo, debe encontrar ID de tarea para el estímulo.
- Tabla de 64 posiciones con ID de tareas (ID de 0 a 64)

# Imagen de la Salida

- La imagen de la salida es una matriz de 5 filas para leds de status.
- El resto se distribuye según conveniencia de armado.
- Por ejemplo salidas duplicadas para leds que indican lo mismo en tablero central y remoto, por razones de potencia.
- Se armarán funciones para el manejo de estos leds.
- La ubicación resultará independiente.

# CUIDADO

- Se presenta un diseño preliminar.
- Requiere de los grupos:
  - Revisión de errores.
  - Mejoras.
  - Mayor claridad.
- Imaginen primera reunión de diseño.
- Definición de Evento:
  - Disparo de un timer.
  - Programado internamente por el sistema.

# Manejo de Tiempos de Eventos.

- La lógica del sistema maneja varios eventos.
- Estos eventos se establecen para que se disparen en un lapso determinado de tiempo.
  - Basta discriminar décimas de segundos.
  - Si se emplean 2 bytes, se cuenta hasta 65536.
- El sistema llevará un “reloj” circular que mide hasta 65536 décimas de seg. (aprox 1,8 hs).
- Se empleará una cola de eventos: EVENTOS.
  - Contiene
    - Identificador del evento (un número).
    - Identificador de la tarea a que corresponde el evento.
    - Valor del “reloj” a que se disparará

# EVENTOS

- Cada tarea maneja numeración propia de eventos. Equivalen a timers en software.
  - ¿Ventajas de esta numeración?
- Cola ordenada por vencimiento.
  - Adelante los que vencen primero.
  - El orden se maneja en módulo 65536.
- Operaciones en la cola (funciones):
  - Insertar evento.
  - Eliminar evento (desactivarlo).
  - Quitar eventos que vencen.
- Qué hilos intervienen.
  - INT incrementa el RELOJ (cada 33 refrescados = 99,9 ms) – equilibrar tiempo todas las INT.
  - ENTRADA

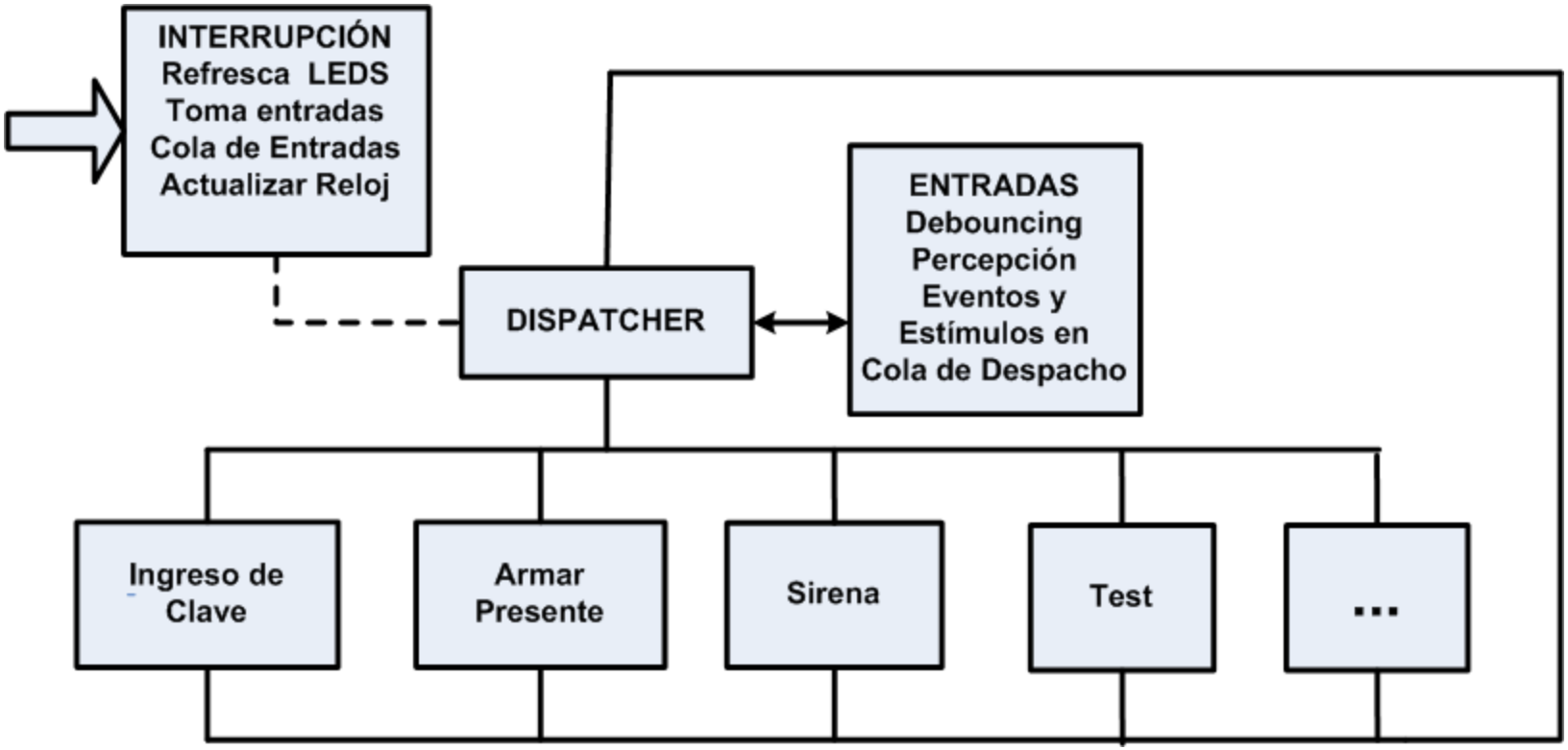
# Descripción ENTRADA

- Detecta estímulos nuevos y los incorpora a la cola de estímulos.
  - Cada ítem contiene el identificador del estímulo y el número de tarea que se encargará del mismo al menos.
- Maneja una cola de eventos.
  - Son futuros vencimientos de distintos timers.
  - Cada tarea maneja sus timers y su numeración.
  - Las tareas convocan a una función de este módulo para encolar el evento de su timer.
  - El evento contiene número de tarea.
- Retira eventos, encola estímulos y eventos.
  - Como **mensajes** en una cola de DESPACHO.
  - Cada mensaje tiene un formato determinado.
- Aborta eventos: los elimina de la cola.

# Descripcion Mensajes

- Formato: tipo, id_tarea, id.
  - Tipo:
    - 1 – evento.
    - 2 – estímulo de flanco ascendente.
    - 3 – ampliación futura – no se usa.
    - 0 – comunicación entre tareas vía dispatcher.
  - Id_tarea: Número de la tarea para eventos o estímulos
  - Id_t (timer):
    - Número de timer en tipo1.Cada tarea maneja sus números de timers en forma independiente.
    - En tipo 2 – no se usa.

# Esquema Gral del Hw.: Hilos.




En Dispatcher hay un salto múltiple, esto es solo un esquema Parcial para uso didáctico..

Para los eventos se salta primero a la tarea y allí al evento por id

# Dispatcher

- Desencola Mensaje de Cola de Despacho.
- En función del mensaje se salta a la tarea.
  - Empleando una tabla de hasta 64 posiciones en ROM.
  - Con las direcciones de las tareas a convocar.
- La tarea discrimina
  - Estímulo y lo maneja con su lógica.
  - En caso de evento salta al que corresponde.
  - Si son muchos eventos puede usar una tabla de saltos adicional.
- Más adelante veremos más detalles.

# Capas

1. Interrupcion - Background.
2. Entrada – Sw de Base.
3. Aplicacion. (Dispatcher + actividades)

- Capa 3 y 2 se comunican mediante funciones.
  - Minimo de variables compartidas.
- ¿Cómo se comunica capa 1? ¿con quién?
- Cada capa tiene una función de inicialización.
  - Se la llama desde la rutina de inicializac.

# Inicialización

- Estado inicial – todo inactivo.
  - Estado = 1 (alarma desactivada).
  - Dar valor: ME=0, MP estado 1 de ROM.
  - Matrices en RAM – por supuesto.
  - Colas Vacías.Imagen Salida 0. Imagen lógica en 0.
  - Stack Vacío e inicializado.
  - Inicializar puertos P1, P2 y P3. P1 entrada, P2 salida, P3 salida. Salidas en 3' stat<e.
    - P1, P2, P3 – nombres genéricos (B, C, D, ej.)
  - Reloj = 0
  - Inicializar Timer y ponerlo en marcha.
  - ¿Puede comenzar Entrada antes que cola de muestras no esté llena?

# Coordinación: INT - ENTRADA

- Interrupción puede:
  - Ingresar nueva muestra a cola datos circular.
  - Reloj del Sistema
- Y al mismo tiempo ENTRADA puede:
  - Estar procesando cola de datos circular
  - O bien eventos del timer.
  - Eventos del timer no se pierden, si la comparación con el ítem que vence más proximamente es por menor o igual.
- Para que funcione bien:
  - Debe haber una coordinación mediante una variable con el número de muestras en cola.
  - Como ya se explicó en tema anterior.

# Coordinación ENTRADA e INT

- ENTRADA modifica N, variable de comunicación con INT de manera atómica (no interrumpible) – N=núm de muestras
- ¿Por qué?
- ¿Como se hace?: inhabilitar interrupciones.
  - Mientras ENTRADA actualiza N.

- No conviene inhabilitar interrupciones durante toda ENTRADA → perjudica sincronismo INT y además no es necesario.

# ENTRADAS

1. IF N>7 /hay datos a procesar
    1. Calcula ME /matriz de estímulos físicos
    2. N=N-1 (PRIMITIVA)
    3. Convierte ME a lógica, MEL. /Procesa datos
    4. Calcula matriz entradas lógicas percibidas (MELP) y matriz transición lógica percibida(MTLP)
        1. MELP = MEL AND MP /matriz de entradas lógicas percibidos.
        2. MTLP = MTL AND MP /matriz de estímulos que cambiaron a “1”.
    5. If MTLP ≠ 0 → Incorpora estímulos nuevos en Cola de Despacho.
2. Revisa Cola de Eventos e incorpora en Cola de Despacho.
3. FIN

Contiene funciones para proveer servicios a las tareas.

# Relación INT - ENTRADA

...

If N<MAX then

    agregar entrada a cola entradas

    N=N+1

End if

...

# DISPATCHER

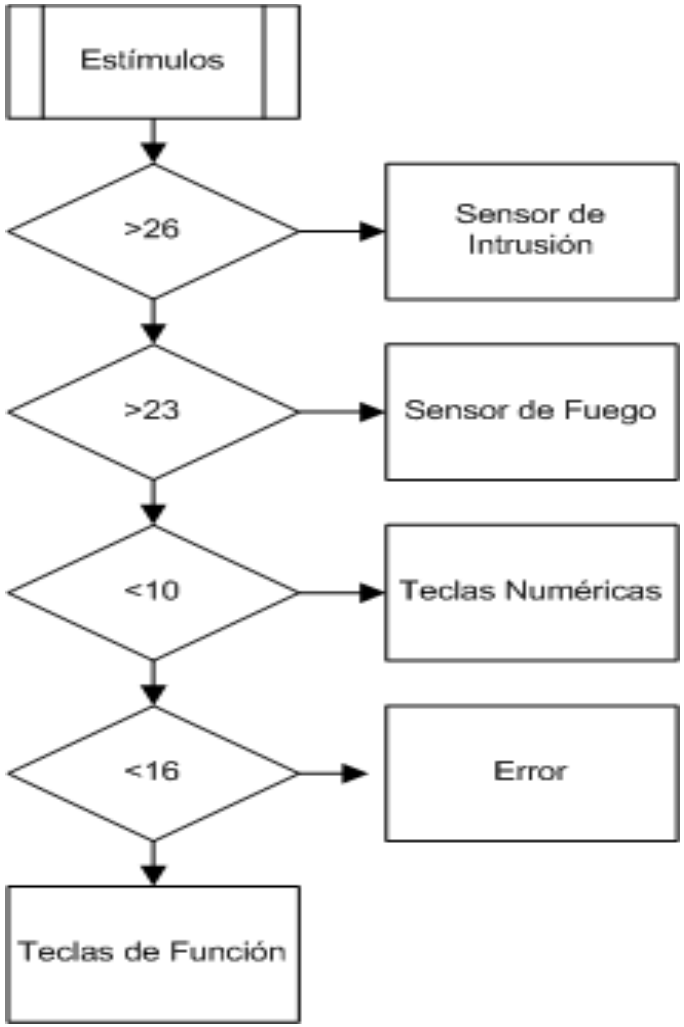
1. Llama a función de ENTRADA
    1. Para procesar muestras.
2. Llama a función de ENTRADA
    1. Para retirar un mensaje.
3. Si hay mensaje DESPACHARLO.
4. Si no hay WAIT.
5. Vaya a 1.

- DISPATCHER se activa luego de una interrupción.

# Tabla de Saltos

- Una entrada por cada tarea diferente (dimensión=cantidad de entradas distintas)
  - Cada tarea se identifica con un ID numérico.
  - Tareas en números sucesivos. ¿Por qué?
  - La tabla está ordenada en forma creciente.
- Da la dirección de la tarea para cada uno de ellos.

# Esquema Tabla de Saltos para Estímulos




Es un esquema general
El salto es directo, no por preguntas.
Hilos por timers más adelante...

# Principales Estructuras de Datos

- Cola de Entradas: 8x8 = 64 Bytes. Circular.
- ME – Matriz de Estado. MEL – lógica.
- MS – Matriz de Salida – se ingresa con funciones.
- MP – Matriz de Percepción.
- MELP – Estados lógicos percibidos
- MTLP – Estímulos percibidos.
- Otras Matrices en RAM y ROM
- 2x64 Bytes en ROM y en RAM – conversión lógica.
- CLAVES en ROM.
- Cola de Despacho. 8 Entradas (mensajes ya descriptos)
- Cola de Timers en espera, ordenada por venc.
  - ID Tarea – para el Dispatcher (de 0 a 64)
  - Timer ID
  - En qué valor de “tic” se dispara.

# Ejemplo valor de “tic”

- Si reloj = 100
  - Y se debe disparar dentro de 100 seg.
  - ¿Qué valor encolar?
- Colocar 1100 en el campo correspondiente y encolar.
- **¿Cómo se disparan eventos?**

# Percepción, ej Forzar Ausente

- Armado forzado Ausente.
  - Revisar si hay un sensor de intrusión activo en el próximo estado → MEPP (P=Próximo).
  - MEPP=MEL AND MPP (MPP – matriz Percepción Próximo Estado, incluye retard.)
  - Anular esos contactos en MPP.
    - MEPF = MPP AND /MEPP (F=Forzado)
    - Quedan en cero los que no se ven.
  - Cuidado con anular sensor de movimiento. Dar tiempo o pedir confirmación.

# Matrices de Percepción (ROM)

- Una para cada modo de armado (2 o 3).
  1. Una para **estado ausente** sin sensores retardados.
  2. Una para **los sensores retardados**
     - Luego se los agrega con OR.
  3. ¿Y para estado presente?.
- Finalmente si es forzado
  - Se realiza el AND recién visto.
- Diferentes combinaciones son posibles.

# Ejercicio

- Se desea incorporar el armado de una zona de la casa.
- Definir la operación para el usuario.
- ¿Cómo implementarlo?
- ¿Se puede hacerlo forzado también?

# ACTIVIDADES

- ¿Matriz de percepción para estados 0 y 1?
- Se verán algunos ejemplos en próximo tema.