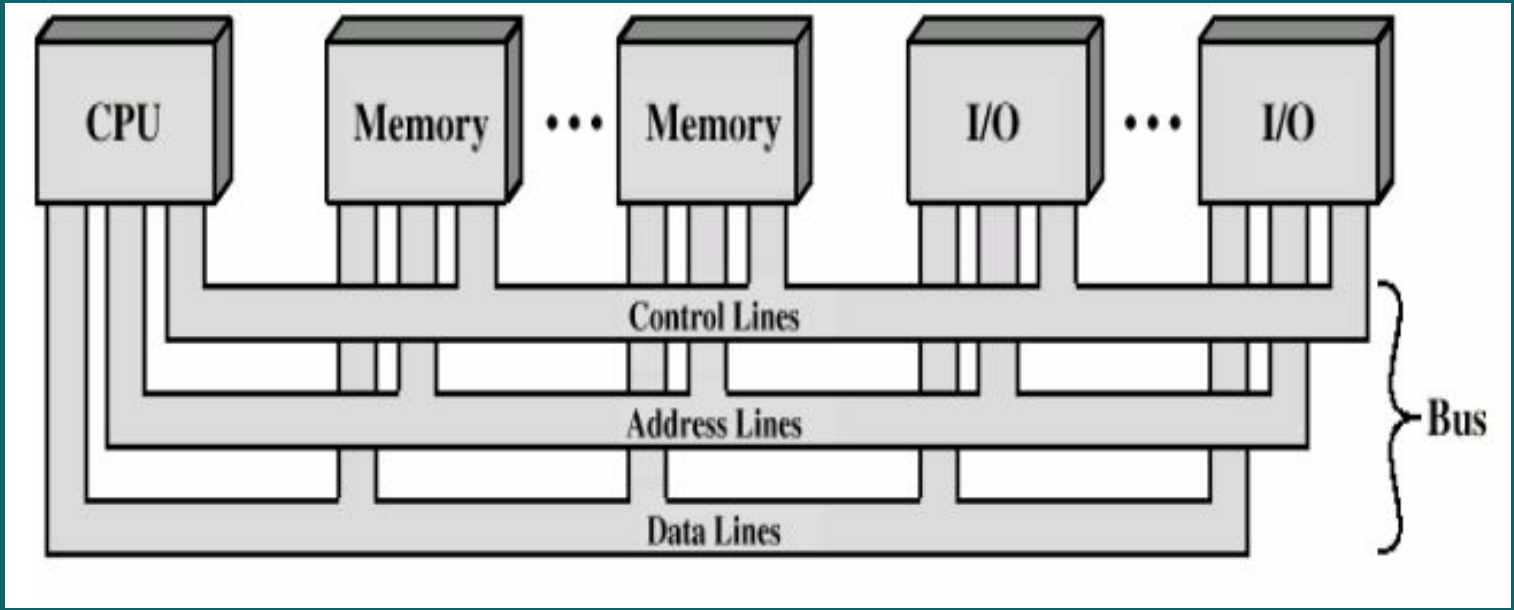
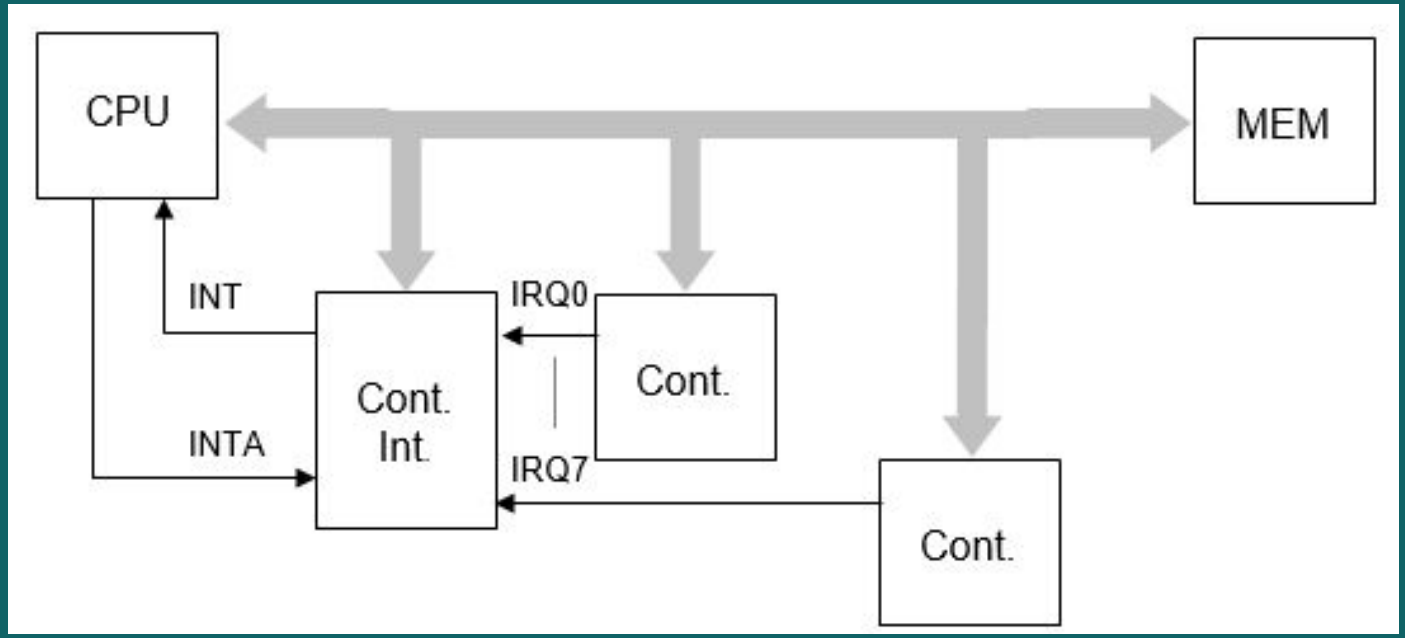
Interrupciones y E/S en 8086

Arquitectura de Computadoras - Práctico 10

Contextualizando

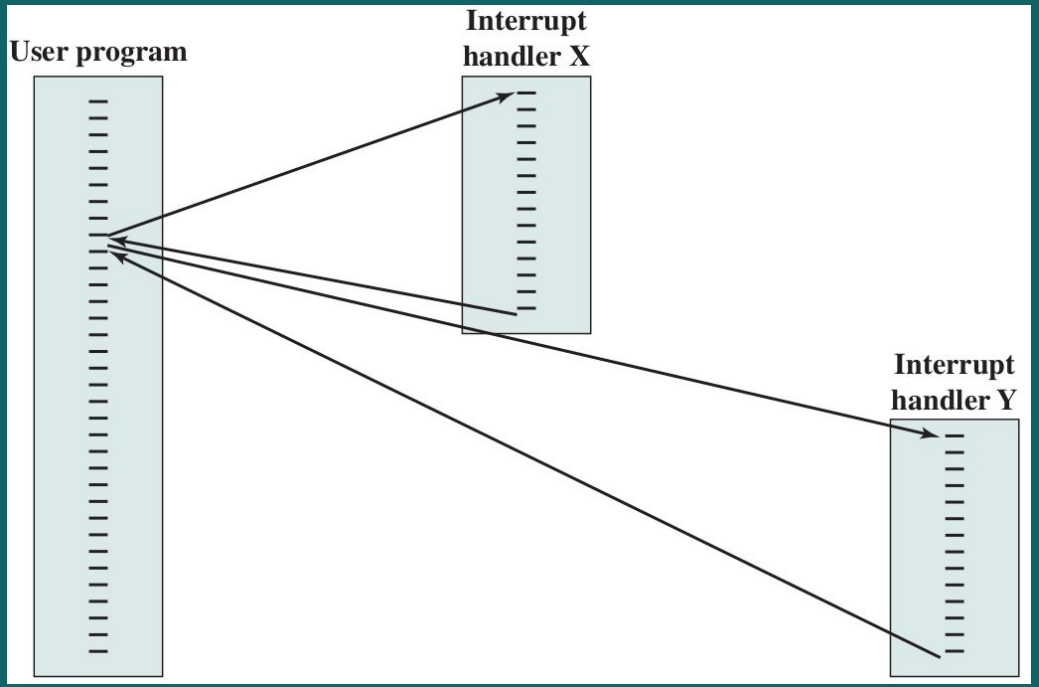


Contextualizando



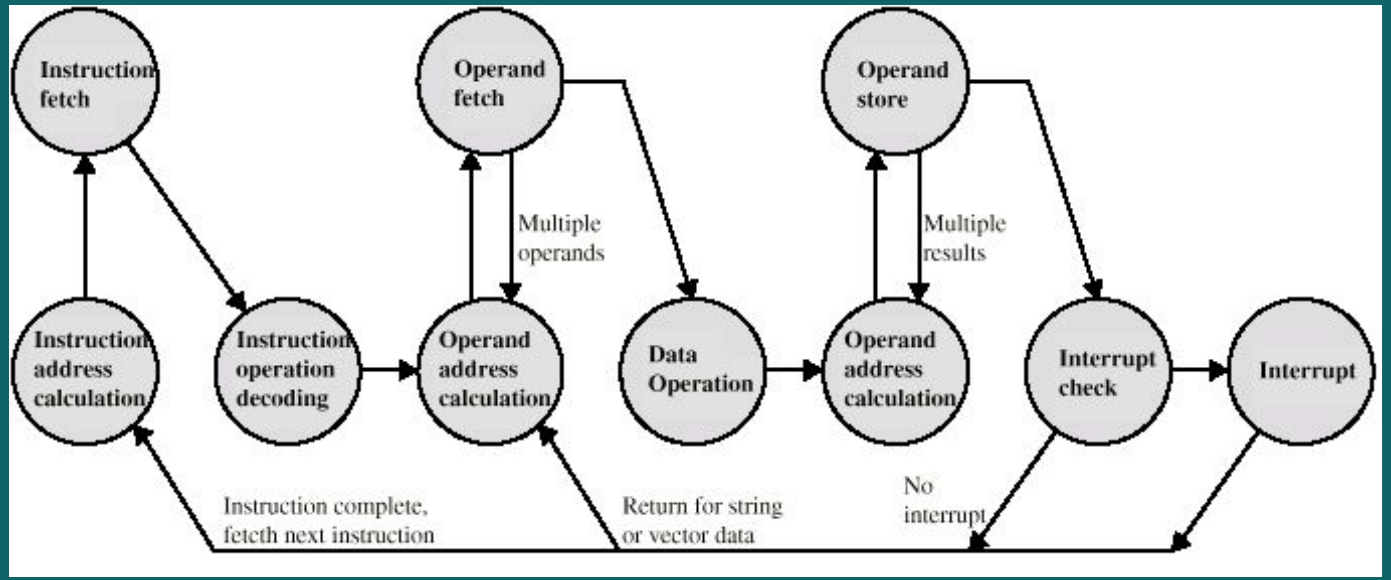
Contextualizando

El manejador de interrupciones

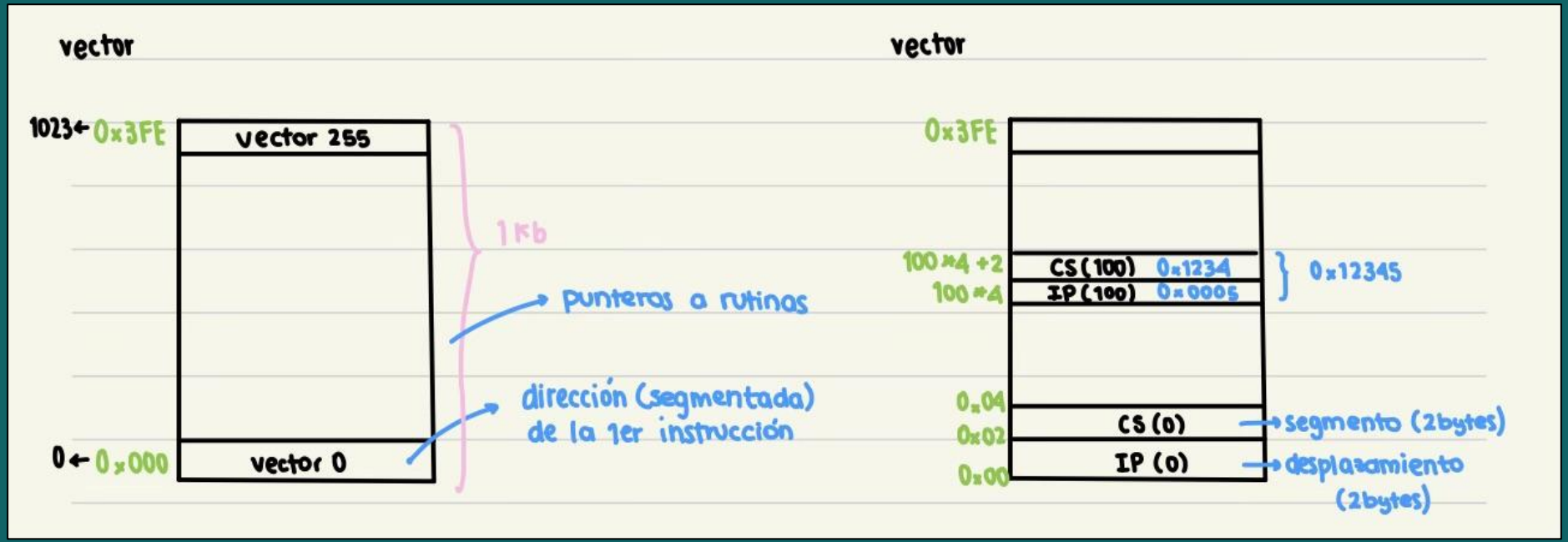


Contextualizando

El ciclo de una instrucción



Vector interrupciones 8086



Aspectos de bajo nivel

- Compilar el alto nivel para ejecutar en un procesador.
- Detalles de 8086
  - in, out, enable y disable
  - iret
  - vector de interrupciones
  - instalación de manejadores de interrupción
  - otras instrucciones pushf, popf
- Instalar interrupciones
- Guardar contexto

Máquinas de Estado

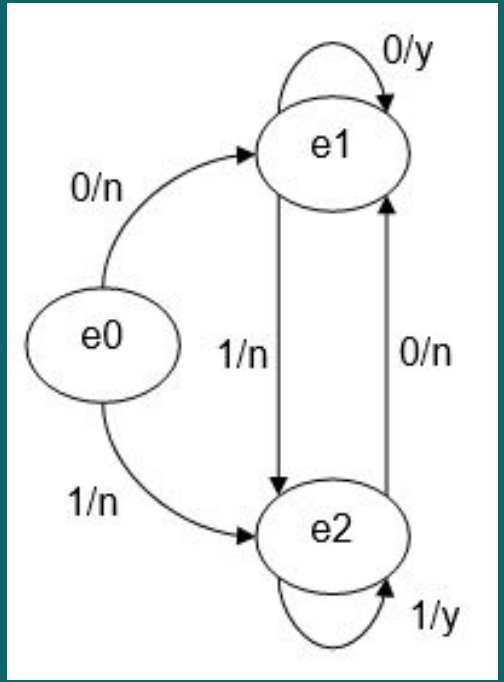
- Las máquinas de estado son un caso particular de Autómatas Finitos Deterministas (AFD), los cuales son un modelo matemático de un sistema con entradas y salidas discretas.

- Son herramientas esenciales en el diseño y control de operaciones secuenciales, siendo las ideales para modelar el manejo de interrupciones; simplificando este proceso y permitiendo al sistema volver a su estado original sin mayores inconvenientes.

Máquinas de Estado

Diagrama de Estado.

Representación gráfica consistente en un grafo orientado; representando con círculos los estados y con arcos orientados las transiciones de estado (la función de transición) y anotando xx...x/yy....y sobre cada transición los valores de la(s) entrada(s) que provocan la transición y el valor correspondiente de la(s) salida(s) (la función de salida).



Ejercicio

Se desea controlar una bomba de agua mediante un microprocesador. La bomba se enciende colocando el valor 1 en el bit 2 del puerto de E/S BOMBA, y se apaga colocando el valor 0 en el mismo bit. Dado que el encendido no es inmediato, existe un **sensor externo que interrumpe** cuando se activa efectivamente la bomba.

Cuando la bomba está activa se debe encender un LED conectado al bit 4 del puerto BOMBA, en otro caso se debe desactivar el LED.

Escribir la rutina de atención a la interrupción y las subrutinas activarBomba() y desactivarBomba().

Modelado del problema

- Modelemos la realidad anterior usando máquina de estados

Alto nivel

¿Quién decide cuándo encender o apagar la bomba?

¿La máquina es dedicada?

¿La rutina *activarBomba ()* debe garantizar que la bomba está encendida luego de su ejecución?

Definición de constantes:

- `#define BOMBA …    ;dirección puerto E/S`
- `#define MASK_LED 16`
- `#define MASK_MOTOR 4`

Alto nivel

```
void main{
     disable ();
     //Instalar interrupciones
     out (BOMBA, 0);
     bom_efectivo=0;
     enable()
}
void activarBomba (){
     out (BOMBA, MASK_MOTOR);
     while bom_efectivo!=1{}
}
void desactivarBomba (){
     out (BOMBA, 0);
     bom_efectivo=0;
}
void interrupcionBomba{
     bom_efectivo=1;
     out (BOMBA, MASK_LED | MASK_MOTOR)
}
```

Asumamos que garantiza encendido

¿Qué cambiaría del código si no fuese necesario dar esta garantía?

¿Cuál es la diferencia para el programador del programa principal?

¿Dónde se define la variable bom_efectiva?

Instalación de interrupciones

```
void main {

  disable()
  //Instalar
  //interrupciones
  OUT (BOMBA,0)
  bom_efectivo=0
  enable ()
```

```
BOMBA EQU puerto
MASK_LED EQU 16
MASK_MOTOR EQU 4
ID_INT_BOMBA EQU valor

bom_efectivo db 0     ;reservo memoria para la variable de
control

main:
  XOR AX, AX
  MOV DX, BOMBA
  OUT DX, AX
  MOV ES, AX
  cli
  MOV ES:[4*ID_INT_BOMBA+2], SEGMENT interrupcionBomba
  MOV ES:[4*ID_INT_BOMBA], OFFSET interrupcionBomba
  sti
```

Ensamblado

```
void activarBomba (){
    out (BOMBA,
MASK_MOTOR);
    while
(bom_efectivo!=1);
}
```

```
ActivarBomba PROC:
    PUSH AX
    PUSH DX
    MOV AX, MASK_MOTOR
    MOV DX, BOMBA
    OUT DX, AX ; Prendo la bomba
    XOR AX, AX
while:       ;Espero interrupción
    CMP AX, byte ptr [bom_efectivo]
    JE while
    POP DX
    POP AX
    RET
ActivarBomba ENDP
```

Ensamblado

```
void desactivarBomba (){
     out(BOMBA, 0)
     bom_efectivo=0
}
```

```
DesactivarBomba PROC:
     PUSH AX
     PUSH DX
     XOR AX, AX
     MOV DX, BOMBA
     OUT DX, AX
     MOV byte ptr [bom_efectivo], 0
     POP DX
     POP AX
     RET
ENDP
```

Ensamblado

```
InterrupcionBomba (){
    bom_efectivo = 1
    out(BOMBA, MASK_MOTOR |
                MASK_LED)
}
```

```
InterrupcionBomba PROC:
    PUSH AX
    PUSH DX
    MOV byte ptr [bom_efectivo], 1
    MOV AX, MASK_MOTOR | MASK_LED
    MOV DX, BOMBA
    OUT DX, AX \\Prendo el led
    POP DX
    POP AX
    IRET
ENDP
```

Máquinas de Estado

Ejemplos de uso.

- Máquina de Moore
  - Semáforo.

Tabla de transición		
Estado actual	**Entrada (temporizador)**	**Estado siguiente**
Verde	Tiempo completo.	Amarillo
Amarillo	Tiempo completo.	Rojo
Rojo	Tiempo completo.	Verde

Tabla de transferencia			
Estado actual	**Luz verde**	**Luz amarilla**	**Luz roja**
Verde	1	0	0
Amarillo	0	1	0
Rojo	0	0	1

- Máquina de Mealy
  - Expendedora de alimentos.

Tabla de transición		
Estado actual	**Entrada**	**Estado siguiente**
Esperando	Moneda = 1	Moneda insertada
Moneda insertada	Moneda = 0	Esperando

Tabla de transferencia		
Estado actual	**Entrada**	**Salida (producto)**
Esperando	Moneda = 1	1 (entrega producto)
Moneda insertada	Moneda = 0	0 (sin producto)