



Arquitectura de Computadoras

Curso 2020 – Prof. Jorge Runco

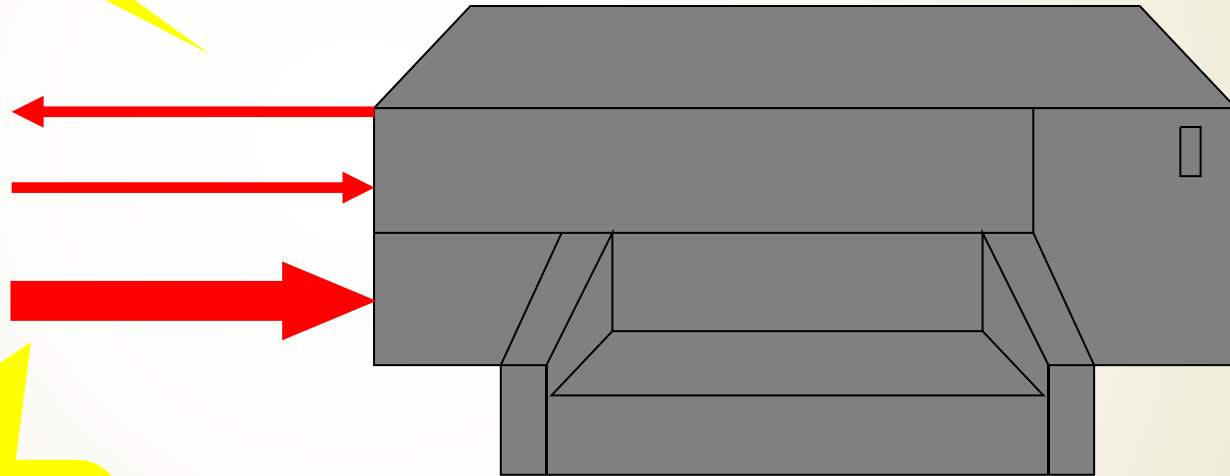
Comunicación CPU – Impresora



Comunicación cpu - impresora

Línea de
BUSY (1 bit)

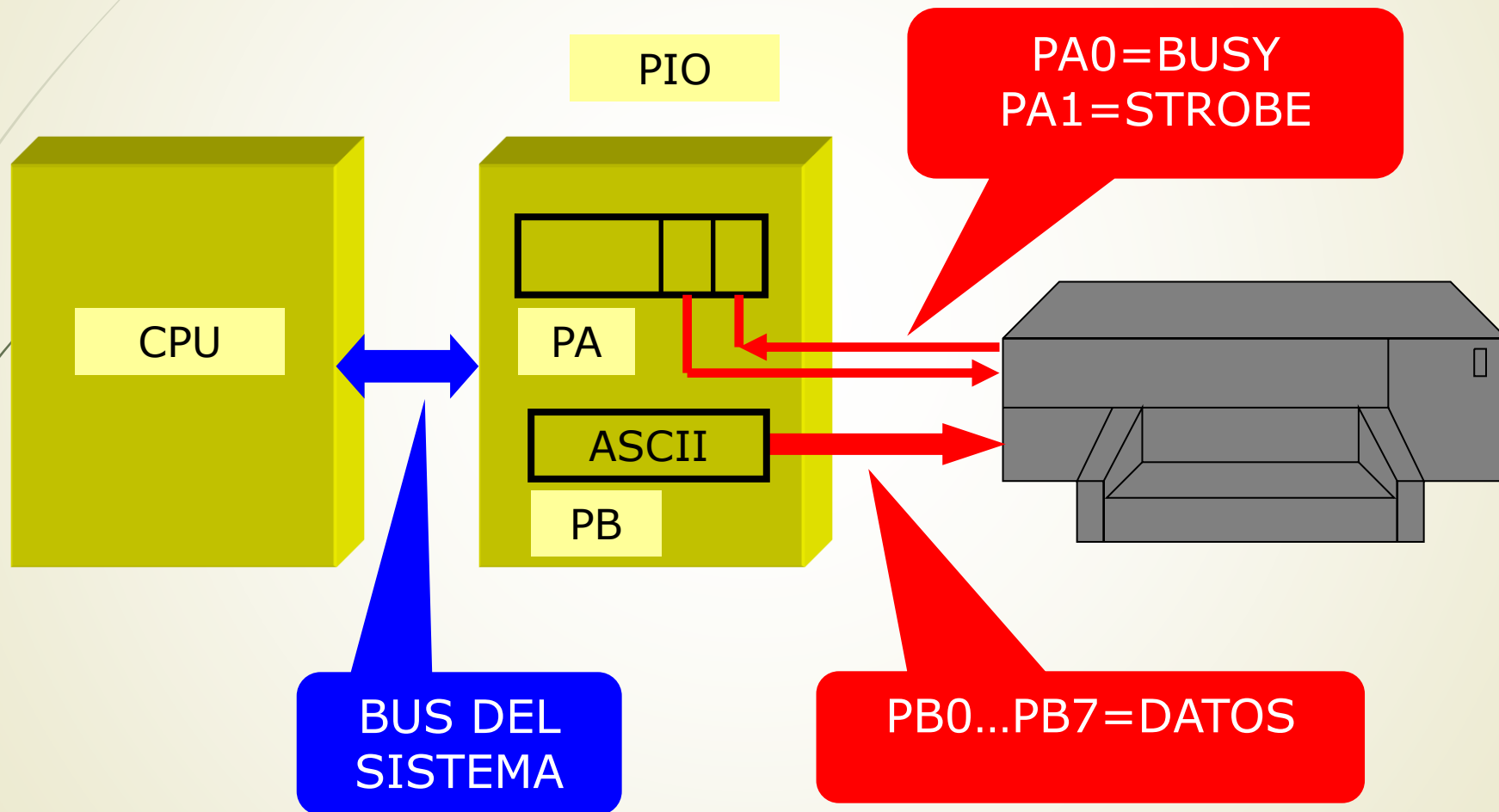
Línea de
STROBE
(1 bit)

Código ASCII del
carácter a
imprimir (8 bits)



- 
- 
- La comunicación con la impresora es a través de 8 líneas entrantes (8 bits) por donde la CPU envía el código ASCII del carácter a imprimir.
 - Otra línea entrante es STROBE por donde la CPU avisa que el carácter ASCII enviado es válido y hay que imprimirlo. Si no está presente esta señal no se lleva a cabo la impresión.
 - Por último una línea saliente BUSY (1 bit) indica cuando la impresora está libre ó ocupada, la impresión se lleva a cabo cuando la impresora está libre. El estado ocupado es cuando está imprimiendo, en el estado libre la CPU envía caracteres para imprimir.

CPU – PIO – IMPRESORA





CPU – PIO – IMPRESORA

- El Puerto B (8 bits) del PIO se conecta a la impresora a las entradas del carácter ASCII, o sea el carácter a imprimir la CPU lo envía a este puerto.
- El bit 0 del Puerto A (PA0) se conecta a la línea de ocupado (BUSY) de la impresora y el bit 1 de este mismo puerto (PA1) se conecta a la entrada de la señal de STROBE.

Secuencia de impresión

- 1) Si la impresora está libre PA0 (bit 0 de PA) en 0, entonces enviar el carácter ascii a imprimir a PB. En caso contrario esperar.
- 2) Generar la señal de STROBE que valida el dato en PB. PA1 está en 0, llevarla a 1 y luego otra vez a 0. Tiene que pasar de 0 a 1 y de 1 a 0, generando así un pulso.
- 3) Volver a 1) si hay más caracteres a imprimir.

2019

```
while (impresora libre & no  
es el último caracter)
```

```
{
```

```
    Enviar el dato
```

```
    Enviar pulso STROBE
```

```
}
```

```
POLL:  IN  AL, PA      }  
        AND  AL, 1     }  
        JNZ  POLL      }  
        MOV  AL, [BX]  }  
        OUT  PB, AL    }  
        MOV  AL, 02H   }  
        OUT  PA, AL    }  
        MOV  AL, 0H    }  
        OUT  PA, AL    }  
        INC  BX  
        DEC  CL        }  
        JNZ  POLL      }
```

Impresora libre ?

Enviar dato

Pulso de Strobe

Último caracter ?

Por consulta de estado (no hay otra opción con el PIO)

PIO EQU 30H

MSJ
ORG 1000H
DB "CONCEPTOS DE"
DB "ARQUITECTURA DE "
DB "COMPUTADORAS"
FIN DB ?

ORG 2000H
MOV AL, 0FDH
OUT PIO+2, AL

MOV AL, 0
OUT PIO+3, AL

IN AL, PIO
AND AL, 0FDH
OUT PIO, AL

Debemos configurar a PA0 como entrada y PA1 como salida. Por

Debemos configurar a PB como salidas. Por eso escribimos en CB

Leo PA y con el AND pongo en 0 el bit1 (strobe) y escribo el nuevo valor en PA.

Con estas operaciones sólo se afecta al bit1. Los demás permanecen inalterados

POLL:

MOV BX, OFFSET MSJ

MOV CL, OFFSET FIN-C

IN AL, PIO

AND AL, 1

JNZ POLL

MOV AL, [BX]

OUT PIO+1, AL

IN AL, PIO

OR AL, 02H

OUT PIO, AL

IN AL, PIO

AND AL, 0FDH

OUT PIO, AL

INC BX

DEC CL

JNZ POLL

INT 0

END

CL=cantidad de

Para saber si la impresora está libre ó no.

PA0=1 impresora ocupada

PA0=0 impresora libre

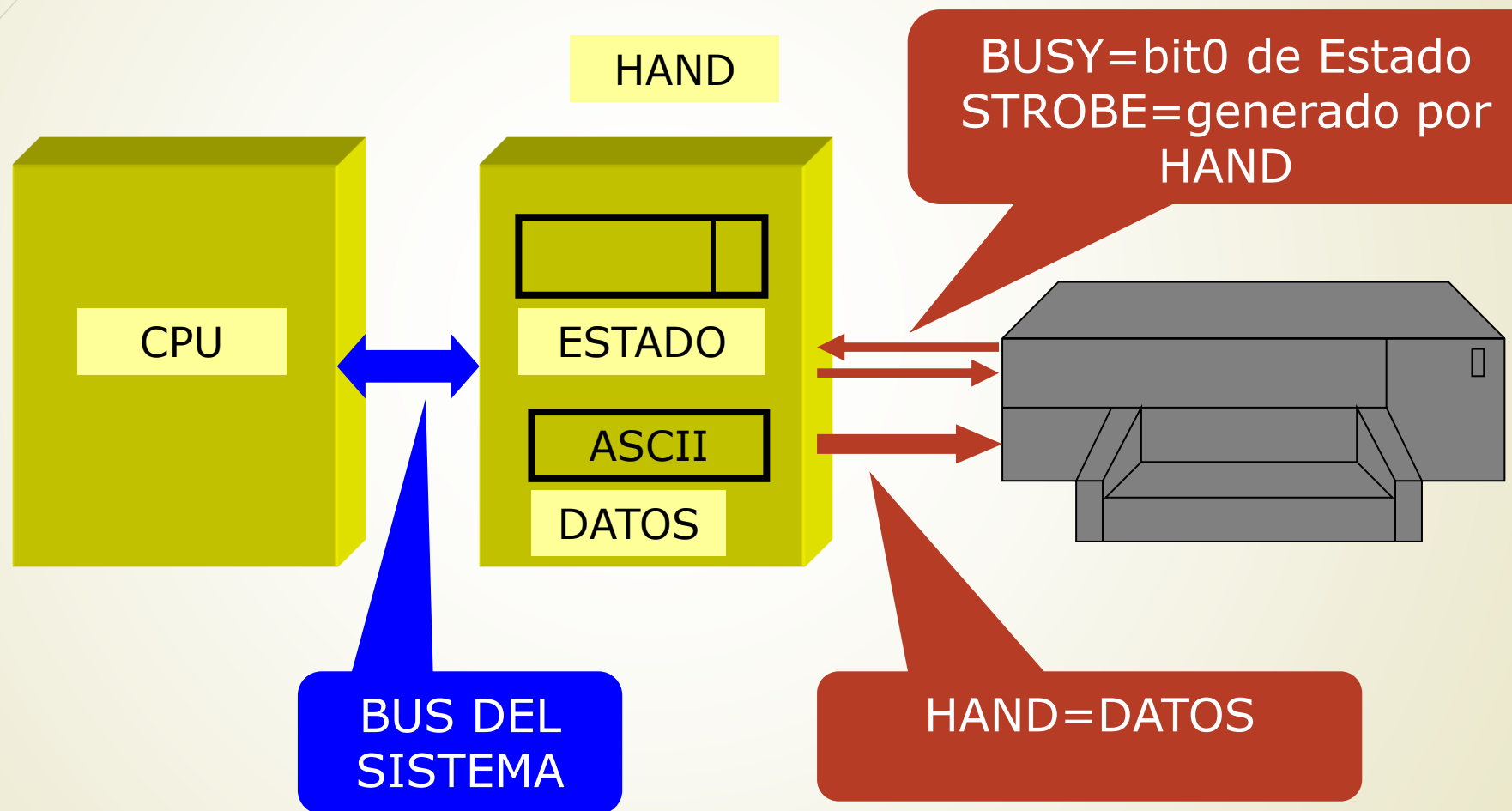
Envía el caracter a la impresora

Generación de pulso de strobe. Valida el caracter enviado a la impresora.

Apunto al siguiente

¿Último caracter?

CPU – HAND – IMPRESORA



CPU – HAND – IMPRESORA

while (impresora libre &
no es el último caracter)

{

Enviar el dato

}

No hay que
generar el pulso
de STROBE. Se
encarga el
HAND.

```
POLL: IN  AL, HAND+1 }  
      AND AL, 1      }  
      JNZ  POLL      }  
      MOV  AL, [BX]   }  
      OUT  HAND, AL   }  
      INC  BX  
      DEC  CL          }  
      JNZ  POLL        }
```

Impresora
libre ?

Enviar
dato

Último
caracter?

Por consulta de estado

```
HAND    EQU    40H

                ORG    1000H
MSJ      DB    "INGENIERIA E "
          DB    "INFORMATICA"
          FIN    DB    ?

                ORG    2000H
          IN AL, HAND+1
          AND AL, 7FH
          OUT HAND+1, AL
          MOV BX, OFFSET MSJ
          MOV CL, OFFSET FIN-OFFSET MSJ

POLL:     IN AL, HAND+1
          AND AL, 1
          JNZ POLL
          MOV AL, [BX]
          OUT HAND, AL
          INC BX
          DEC CL
          JNZ POLL
          INT 0
          END
```

2019

Ej7TP3

```
ORG 2000H
```

```
IN AL, HAND+1
```

Leo el registro de ESTADO

```
AND AL, 7FH
```

Pone en 0 el bit 7 de registro de Estado, para que trabaje por consulta de estado

```
OUT HAND+1, AL
```

```
MOV BX, OFFSET MSJ
```

Apunta al texto a enviar a imprimir (BX) y en CL está el número de caracteres

```
MOV CL, OFFSET FIN-OFFSET MSJ
```

POLL:

```
IN AL, HAND+1
```

```
AND AL, 1
```

El bit0 del registro de ESTADO nos dice si la impresora está libre (0) ó está ocupada imprimiendo (1).

```
JNZ POLL
```

```
MOV AL, [BX]
```

Se envía el carácter a imprimir al registro DATOS del HAND. El HAND se encarga de enviarlo a la impresora y

```
OUT HAND, AL
```

```
INC BX
```

Apunta al siguiente caracter a imprimir

```
DEC CL
```


```
JNZ POLL
```

¿Último caracter a imprimir?

```
INT 0
```

```
END
```

CPU – HAND – IMPRESORA



```
Procedure Interrupt  
{ Envía caracter;  
  Un carácter menos;  
}
```

```
begin  
while (no es el último  
caracter)  
{  
  .....  
}
```

Solo hay que
verificar si es el
último carácter
(programa
principal)

```
POLL:  CMP  CL, 0  
        JNZ  POLL
```

Programa
principal

```
RUT_HND: PUSH  AX  
          MOV   AL, [BX]  
          OUT   HAND, AL  
          INC   BX  
          DEC   CL  
          MOV   AL, 20H  
          OUT   PIC, AL  
          POP   AX  
          IRET
```

Rutina de
interrupción. El
HAND
interrumpe
cuando la
impresora está
libre. Hay que
enviar el
caracter y
decrementar
la cantidad

Interrupción generada por el HAND

2019

Ej8TP3

PIC EQU 20H
HAND EQU 40H
N_HND EQU 10
ORG 40
IP_HND DW RUT_HND
ORG 1000H
MSJ DB "UNIVERSIDAD "
DB "NACIONAL DE LA PLATA"
FIN DB ?
ORG 3000H
RUT_HND: PUSH AX
MOV AL, [BX]
OUT HAND, AL
INC BX
DEC CL
JNZ FINAL
MOV AL, 0FFH
OUT PIC+1, AL
FINAL: MOV AL, 20H
OUT PIC, AL
POP AX
IRET

ORG 2000H
MOV BX, OFFSET MSJ
MOV CL, OFFSET FIN-OFFSET MSJ
CLI
MOV AL, 0FBH
OUT PIC+1, AL
MOV AL, N_HND
OUT PIC+6, AL
MOV AL, 80H
OUT HAND+1, AL
STI
LAZO: CMP CL, 0
JNZ LAZO
IN AL, HAND+1
AND AL, 7FH
OUT HAND+1, AL
INT 0
END


```
RUT_HND:  ORG 3000H
           PUSH AX
           MOV AL, [BX]
           OUT HAND, AL
           INC BX
           DEC CL
           JNZ FINAL
           { MOV AL, 0FFH
             OUT PIC+1, AL
             { MOV AL, 20H
               OUT PIC, AL
               POP AX
               IRET
```

Escribe en IMR

FINAL:

Se ejecutan siempre para avisarle al PIC que terminó la ejecución de la subrutina.

Salva AX en la pila

Envía el caracter al HAND

Apunta al caracter siguiente

¿Último caracter?

Sí. Sigue por acá y deshabilita interrupciones en el PIC. IMR=FFH

NO. Sigue por acá

```
ORG 2000H
```

```
MOV BX, OFFSET MSJ
```

```
MOV CL, OFFSET FIN-OFFSET MSJ
```

```
CLI
```

```
MOV AL, 0FBH
```

```
OUT PIC+1, AL
```

```
MOV AL, N_HND
```

```
OUT PIC+6, AL
```

```
MOV AL, 80H
```

```
OUT HAND+1, AL
```

```
STI
```

```
CMP CL, 0
```

```
JNZ LAZO
```

```
IN AL, HAND+1
```

```
AND AL, 7FH
```

```
OUT HAND+1, AL
```

```
INT 0
```

```
END
```

LAZO:

BX apunta al mensaje y
en CL la cantidad de

Inhibimos interrupciones a nivel CPU

bit2=0 en la máscara (IMR) para habilitar
int2 producida por el HAND.

Escribe en INT2 el lugar seleccionado en
la tabla de vectores

Pone en 1 el bit 7 del registro de Estado,
para que trabaje por interrupciones.

Habilitamos interrupciones a nivel CPU

¿Último carácter?

Pone en 0 el bit 7 del registro de
Estado para terminar de trabajar por
interrupciones.



XXXXXXXXX
AND 111111101
xxxxxx0x