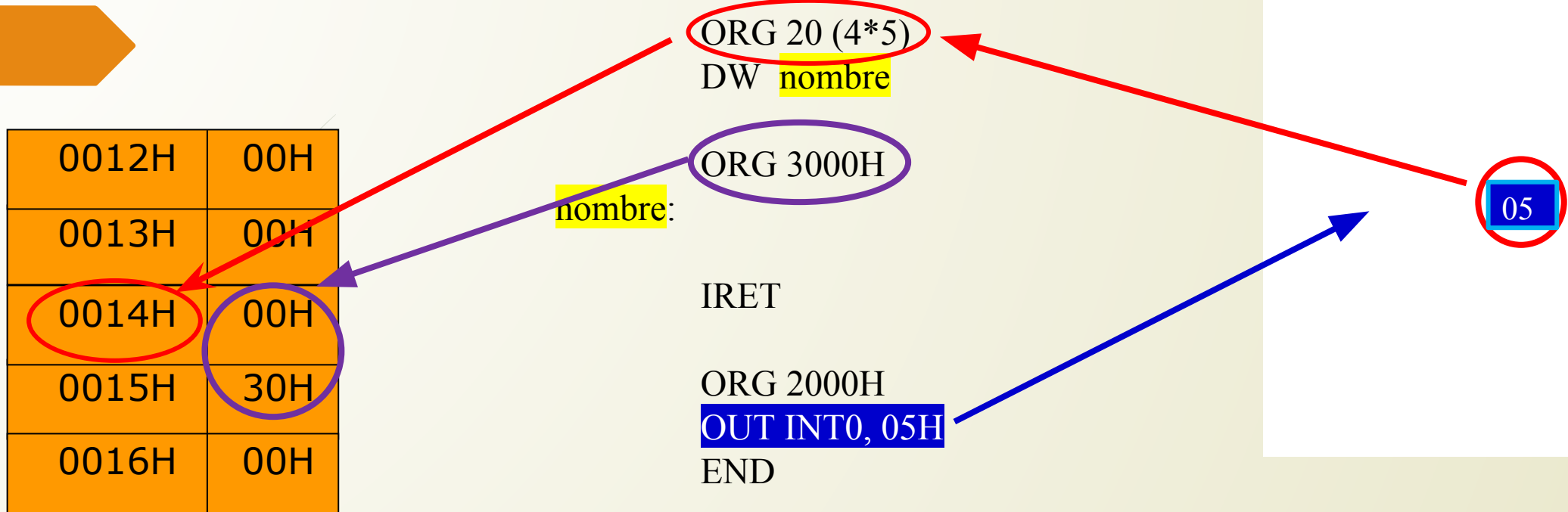




Arquitectura de Computadoras

Curso 2020 – Prof. Jorge Runco

Ejemplos: interrupciones por software y hardware



The diagram illustrates the relationship between memory addresses and assembly code. On the left, a table shows memory addresses from 0012H to 0016H and their corresponding values. Arrows indicate the following: a red arrow from the circled address 0014H to the instruction 'ORG 20 (4*5)'; a purple arrow from the circled value 00H to the label 'nombre:'; a red arrow from the circled value 30H to the instruction 'ORG 20 (4*5)'; and a blue arrow from the instruction 'OUT INT0, 05H' to a circled value 05.

0012H	00H
0013H	00H
0014H	00H
0015H	30H
0016H	00H

ORG 20 (4*5)

DW nombre

ORG 3000H

nombre:

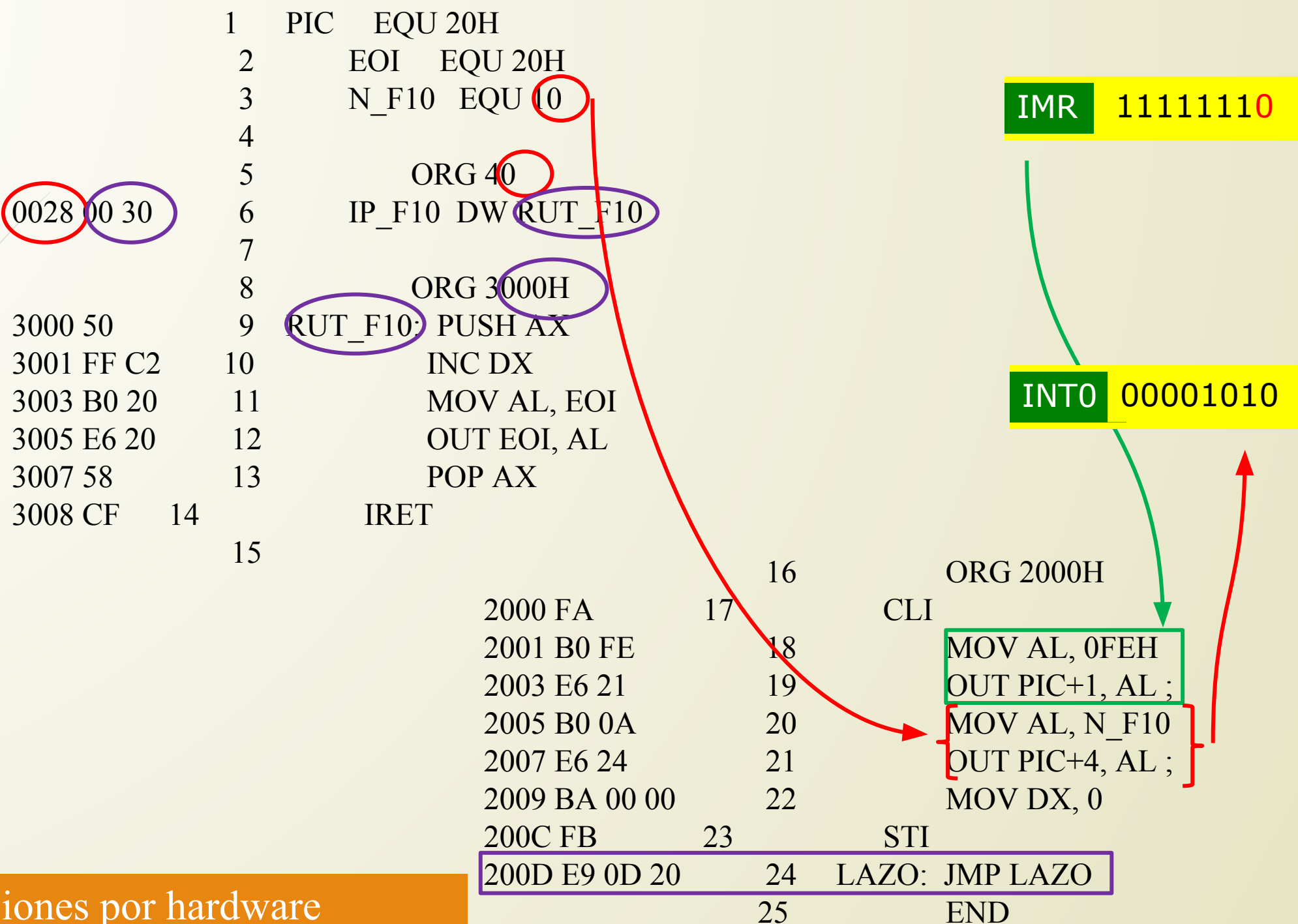
IRET

ORG 2000H

OUT INT0, 05H

END

05



Ejemplo interrupciones por hardware

EJ5TP2

PILA

7FFE H	10 H
7FFF H	20 H
8000 H	00 H

SUBROUTIN

3000 H	50 H
3001 H	FF H
3002 H	C2 H
3003 H	B0 H

PRINCIPAL

200C H	00 H
200D H	E9 H
200E H	0D H
200F H	20 H
2010 H	00 H

CPU

4x10=40=28H

SP 7FFE H

IP 3000 H

VECTORES (MEM)

0027 H	00 H
0028 H	00 H
0029 H	30 H
002A H	00 H

PIC

1111111 0

IMR

INT0 10

INT1

INT2

INT3

F10

INT

INT

A

BUS DE DATOS

EJ5TP2

```

ORG 1000H
NUM DB ?
MSJ1 DB "CARACTER NO VÁLIDO"
FIN1 DB ?
MSJ2 DB "CARACTER VÁLIDO"
FIN2 DB ?

```

```

ORG 3000H
ES_NUM: MOV AH, [BX]
        SUB AH, 30H
        CMP AH, 10
        JNC NO_ES
        MOV AH, 0FFH
        JMP FIN
NO_ES: MOV AH, 00H
FIN: RET

```

```

NO_ES_NUM: MOV BX, OFFSET MSJ1
            MOV AL, OFFSET FIN1 - OFFSET MSJ1
            INT 7
AFUERA:    HLT
END

```

```

ORG 2000H
MOV BX, OFFSET NUM
INT 6
CALL ES_NUM
CMP AH, 00H
JZ NO_ES_NUM
MOV BX, OFFSET MSJ2
MOV AL, OFFSET FIN2 - OFFSET MSJ2
INT 7
JMP AFUERA

```

Modificar el programa anterior agregando una subrutina llamada ES_NUM que verifique si el caracter ingresado es realmente un número. De no serlo, el programa debe mostrar el mensaje "CARACTER NO VALIDO". La subrutina debe recibir el código del caracter por referencia desde el programa principal y debe devolver vía registro el valor 0FFH en caso de tratarse de un número o el valor 00H en caso contrario. Tener en cuenta que el código del "0" es 30H y el del "9" es 39H.

EJ5TP2

ORG 2000H

MOV BX, OFFSET NUM
INT 6

CALL ES_NUM

CMP AH, 00H

JZ NO_ES_NUM

MOV BX, OFFSET MSJ2

MOV AL, OFFSET FIN2 - OFFSET MSJ2

INT 7

JMP AFUERA

NO_ES_NUM: MOV BX, OFFSET MSJ1

MOV AL, OFFSET FIN1 - OFFSET MSJ1

INT 7

AFUERA: HLT

END

En BX se carga con la dirección donde se...

El caracter ingresado por teclado se almacenó en NUM (BX). Ahora llamamos a la rutina que comprueba si es un número. Si es un número AH=FF y si no AH=00

EJ5TP2

ORG 3000H

ES_NUM: MOV AH, [BX]

SUB AH, 30H

CMP AH, 10

JNC NO_ES

MOV AH, 0FFH

JMP FIN

NO_ES: MOV AH, 00H

FIN: RET

BX 1000H

AX 28 00H

AX

Supongamos que el carácter ingresado es un 8. Entonces

Si es un número el resultado de la cuenta estará entre 0 y 9. Al restar 10 habrá borrow (Carry=1). Si no es un número no hay borrow (Carry=0).

Estas 4 instrucciones funcionan como un if ☐ if (no hay carry) then AH=00 else AH=FF

EJ5TP2

ORG 3000H

ES_NUM: MOV AH, [BX]

SUB AH, 30H

CMP AH, 10

JNC NO_ES

MOV AH, 0FFH

JMP FIN

NO_ES: MOV AH, 00H

FIN: RET

Si hay carry no salta (es un número).

Hace el else AH=FF. El JMP hace falta para no ejecutar el then también.

Si no hay carry salta a NO_ES (no es un número).

AX FF 00H

AX 00 00H

Estas 4 instrucciones funcionan como un if ☐ if (no hay carry) then AH=00 else AH=FF


```
ORG 2000H
MOV BX, OFFSET NUM
INT 6
CALL ES_NUM
CMP AH, 00H
JZ NO_ES_NUM
MOV BX, OFFSET MSJ2
MOV AL, OFFSET FIN2 - OFFSET MSJ2
INT 7
JMP AFUERA
NO_ES_NUM:  MOV BX, OFFSET MSJ1
            MOV AL, OFFSET FIN1 - OFFSET MSJ1
            INT 7
AFUERA:    HLT
END
```

EJ5TP2

```
ORG 2000H
MOV BX, OFFSET NUM
INT 6
CALL ES_NUM
CMP AH, 00H
JZ NO_ES_NUM
MOV BX, OFFSET MSJ2
MOV AL, OFFSET FIN2 - OFFSET MSJ2
INT 7
JMP AFUERA
NO_ES_NUM: MOV BX, OFFSET MSJ1
MOV AL, OFFSET FIN1 - OFFSET MSJ1
INT 7
AFUERA:   HLT
END
```

Es un if que analiza el valor devuelto por la subrutina en AH. Llama a INT 7 para mostrar un cartel en la pantalla. Recordemos que antes de llamar a la función que imprime en pantalla, los parámetros pasados son BX= dirección donde empieza el mensaje y AL= cantidad de elementos a mostrar.