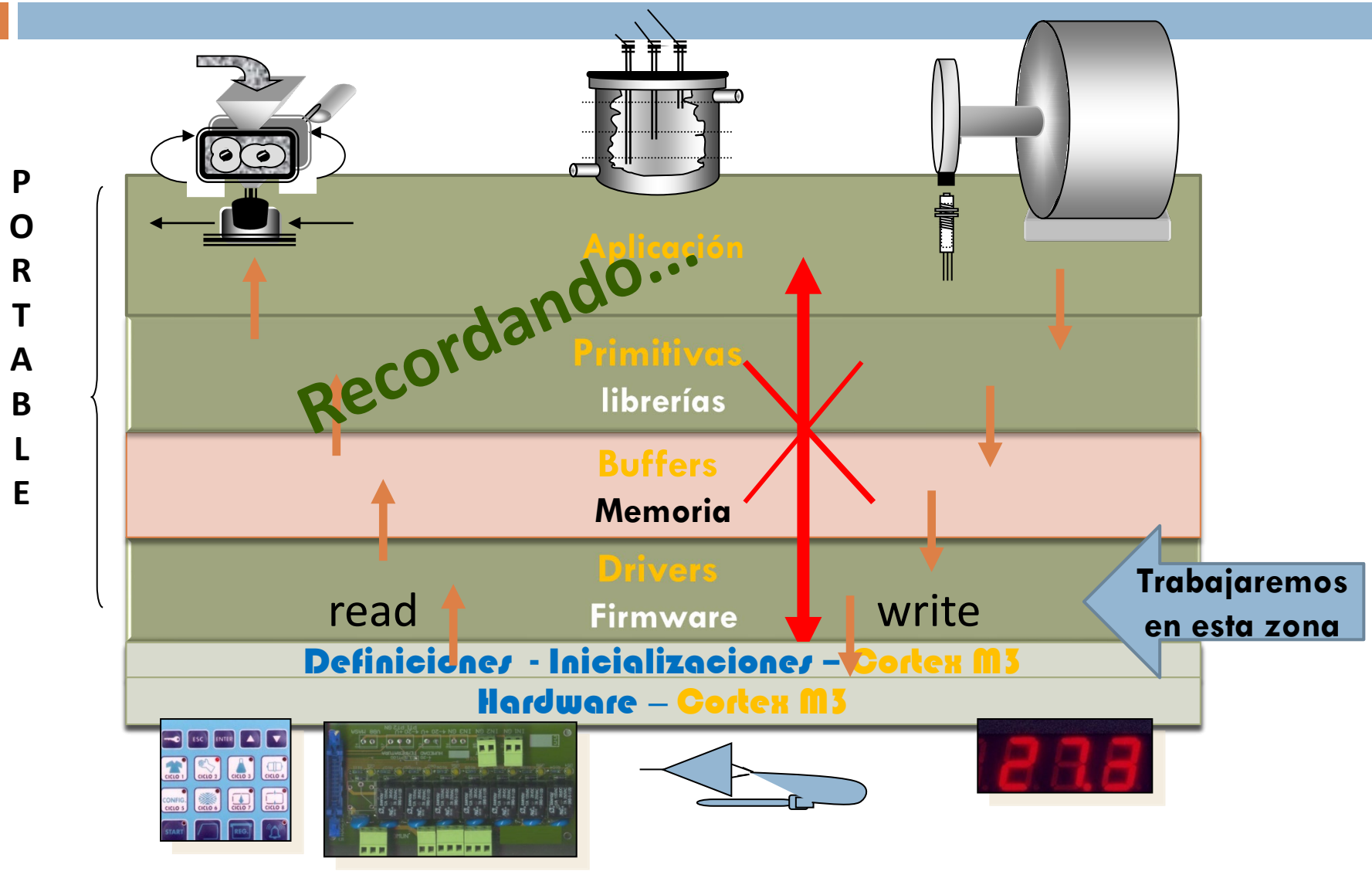
INTERRUPCIONES

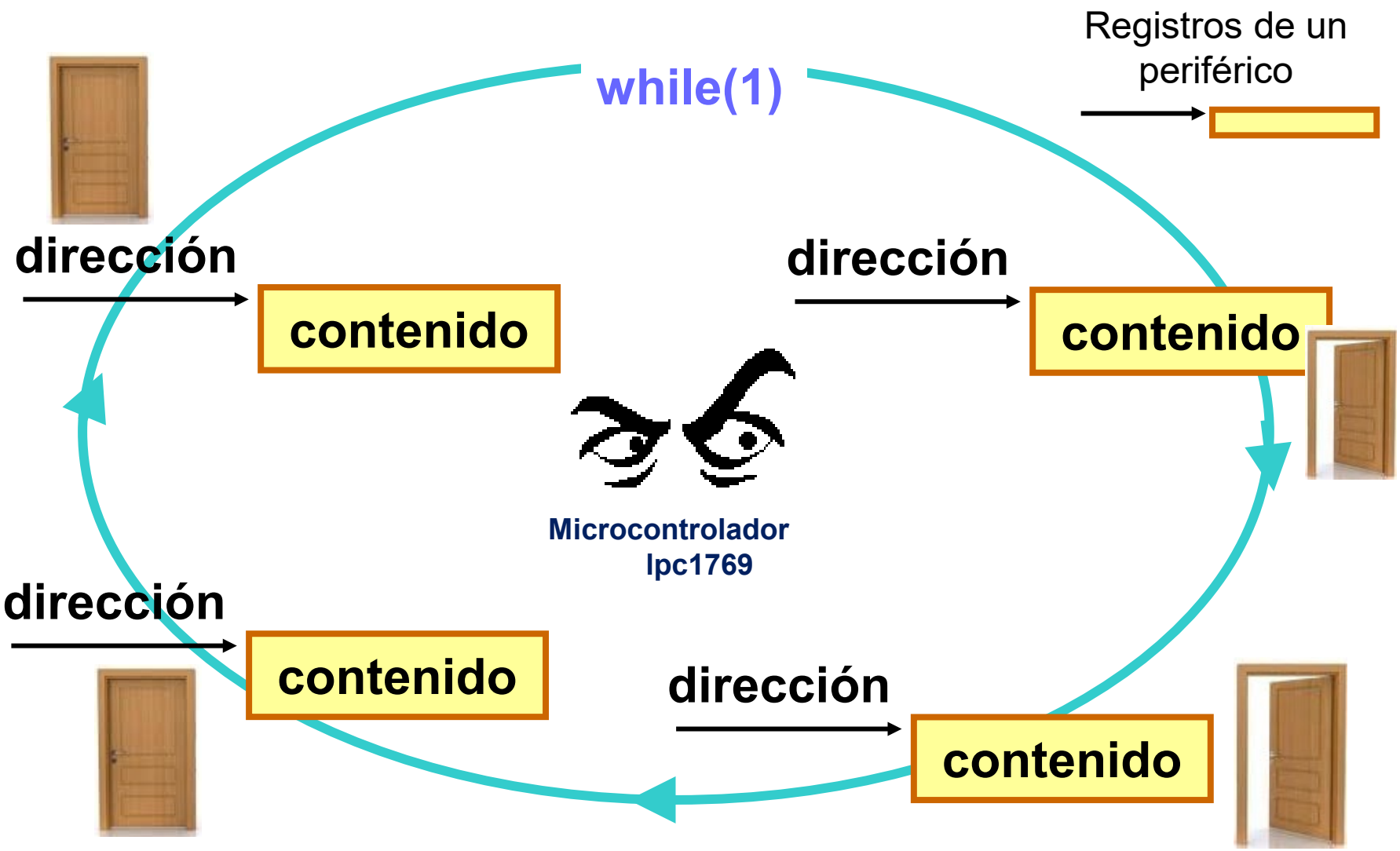
Informática II – Departamento de Electrónica

El metodo

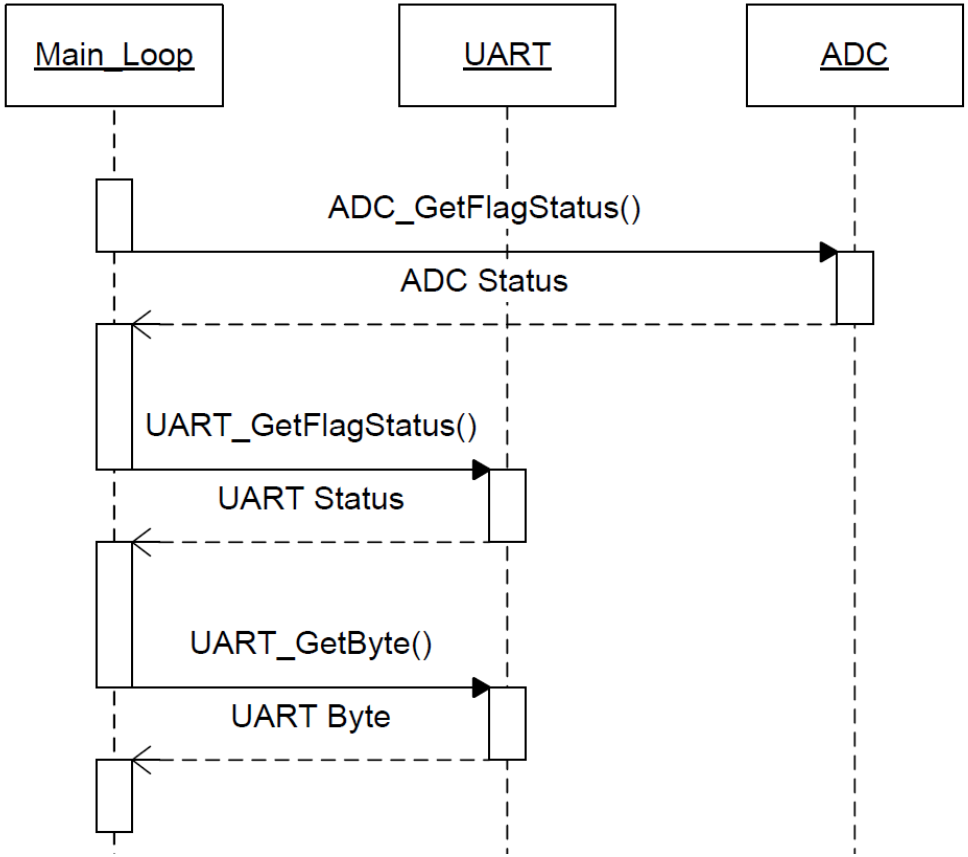

PORTABLE
Aplicación
Recordando...
Primitivas
librerías
Buffers
Memoria
Drivers
Firmware
read
write
Definiciones - Inicializaciones – Cortex M3
Hardware – Cortex M3
Trabajaremos en esta zona

Drivers → Scheduler

Encuesta - Pooling

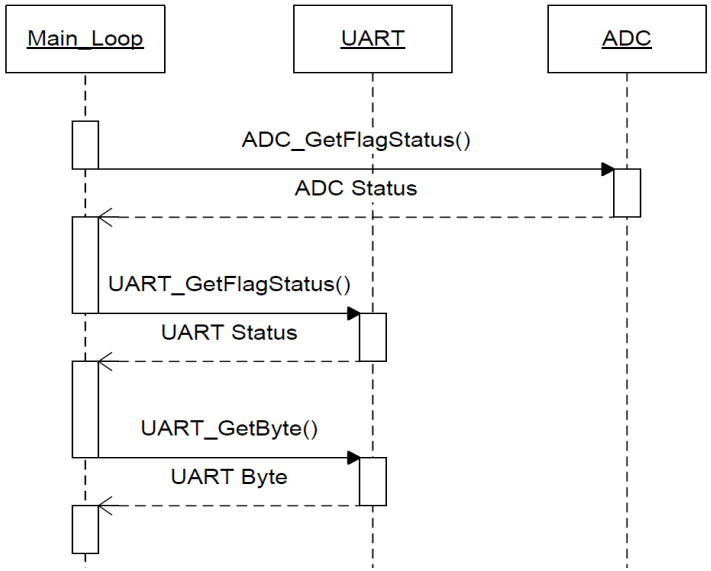


Programando SIN usar interrupciones: pooling



➔ The **`main()`** function executes all peripheral calls in a fixed sequence

Encuesta



La atención de cada periférico es:

¡¡SINCRONICA CON LA APLICACIÓN!!

Interrupciones - Introducción

Voy al cine con mi hija

Secuencia planificada

→Llegamos al cine

→Saco las entradas

→Compro Pochoclos

→Comemos Pochoclos

→Esperamos para entrar

→Entramos a la sala

→Busco la butaca

→Nos sentamos

→Vemos la película

→Nos vamos

→Trago pochoclo

→Atiendo

→Conversación

→Cuelgo

¿Que estaba haciendo yo ?

Interrupciones - Introducción

→Llegamos al cine

→Saco las entradas

→Compro Pochoclos

→Comemos Pochoclos

→Esperamos para entrar

→Entramos a la sala

→Busco la butaca

→Nos sentamos

→Vemos la película

→Nos vamos

Interrupciones - Introducción

¿Que hice cuando llego la interrupción ?

→Tragué pochoclo
→Terminé de pagar
→Me acomodé en el asiento

Concluí con lo que estaba haciendo en ese momento

¿Y antes que eso ?

Tome nota de con que seguir cuando termine de hablar

… y luego

¡Atendí el teléfono!

¿Por último ?

Continué con lo que estaba haciendo

Interrupciones - Introducción

¡Mas interrupciones !!!... Y llegan juntas

→Llegamos al cine

→Saco las entradas

→Compro Pochoclos

→Comemos Pochoclos

→Esperamos para entrar

→Entramos a la sala

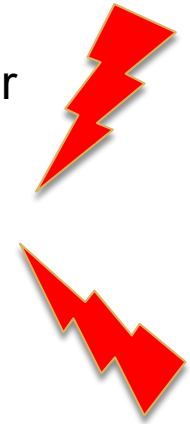
→busco la butaca

→Nos sentamos

→Vemos la película

→Nos vamos

Mi mujer para ver si conseguimos entradas



Papi .. me das agua

¡Que hago !!!?

Interrupciones - Introducción

¿¡Que hago !!!?

→*Establezco prioridades*

→*Atiendo una por vez*

Interrupciones - Introducción

Muchas interrupciones juntas ***¡cuando comenzó la película !!!***

Mi mujer para ver si llegamos bien

Mi mujer para ver si el nene trajo abrigo

Mi mujer para saber a que hora llegamos

→Llegamos al cine

→Saco las entradas

→Compro Pochoclos

→Comemos Pochoclos

→Esperamos para entrar

→Entramos a la sala

→busco la butaca

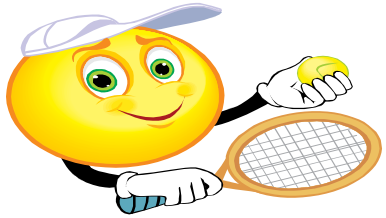
→Nos sentamos

→Vemos la película

→Nos vamos

Cuando salimos, ¿ me compras ?..

Vamos al club el próximo el sábado ?



Vamos mañana al club ?

¡BASTA !!!!!

Interrupciones - Introducción

¡Quiero ver la película tranquilo !!
¡¿Que hago ?!

Bloqueo las interrupciones !!!!

¿y después ?

Las habilito al término de la película

Interrupciones - Introducción

→Llegamos al cine

→Saco las entradas

→Compro Pochoclos

→Comemos Pochoclos

→Esperamos para entrar

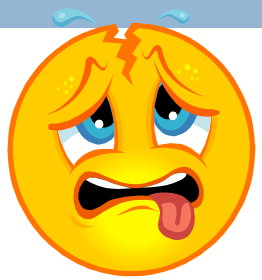
→Entramos a la sala

→busco la butaca

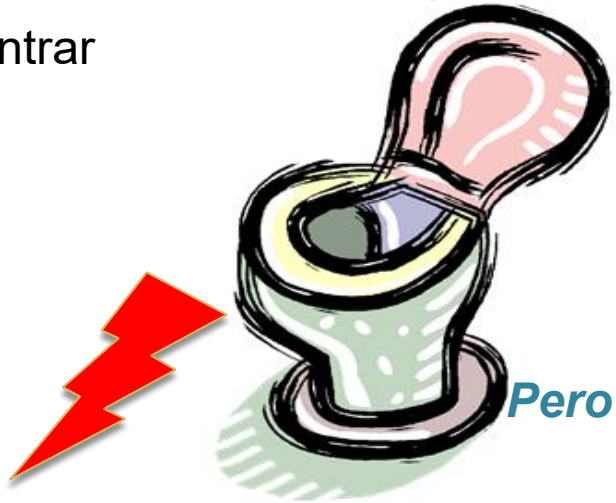
→Nos sentamos

→Vemos la película

→Nos vamos



Algunas interrupciones no se pueden bloquear



Pero que pasa si …

Interrupciones - Resumen

Llega la interrupción y…

→*Termino lo que estoy haciendo*
→*Recuerdo con que continuar*
→*Realizo la tarea que me interrumpió*
→*Continúo con lo que estaba haciendo*

Si existen varias fuentes de Interrupción…

→*Establecemos prioridades*

Las interrupciones pueden ser :

→*Bloqueables (enmascarables)*

→*No Bloqueables (no enmascarables)*

Interrupciones - Resumen

… Y en nuestro programa

Llega la interrupción y…

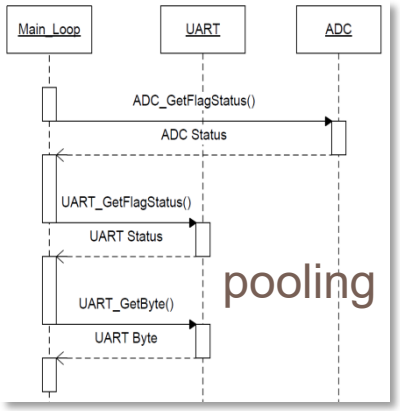
→Se concluye la instrucción en curso
→Se guarda automáticamente la dirección de retorno
→Se guarda automáticamente el PSW

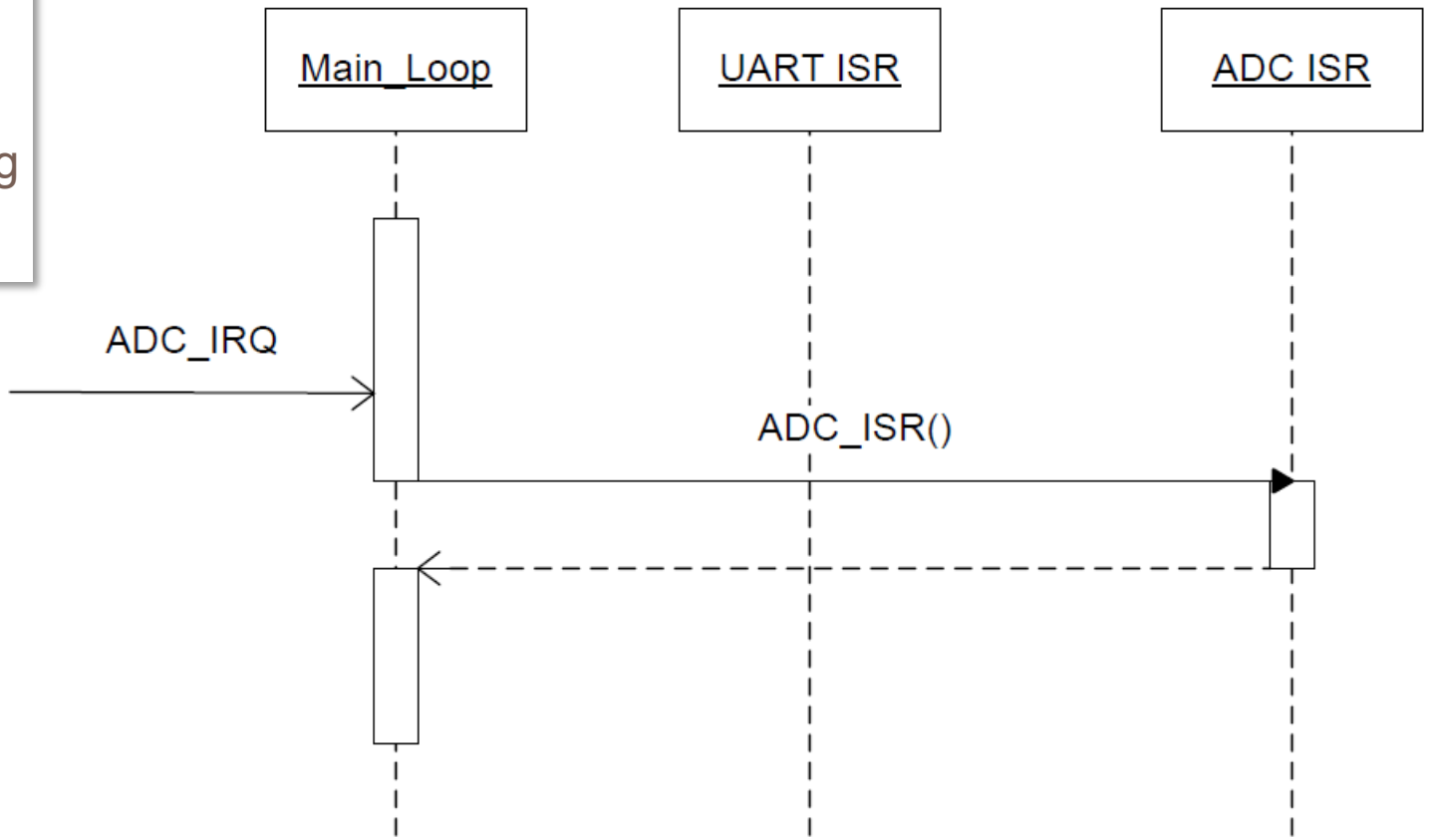
→Atiendo la función de interrupción.

→Se recupera automáticamente el PSW
→Se recupera automáticamente la dirección de retorno
→Retorno al lugar al que me interrumpieron

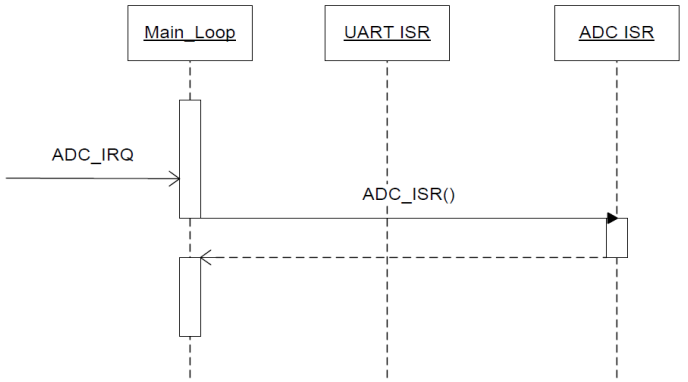
Programando haciendo USO de interrupciones

recordando...





Interrupciones - Resumen

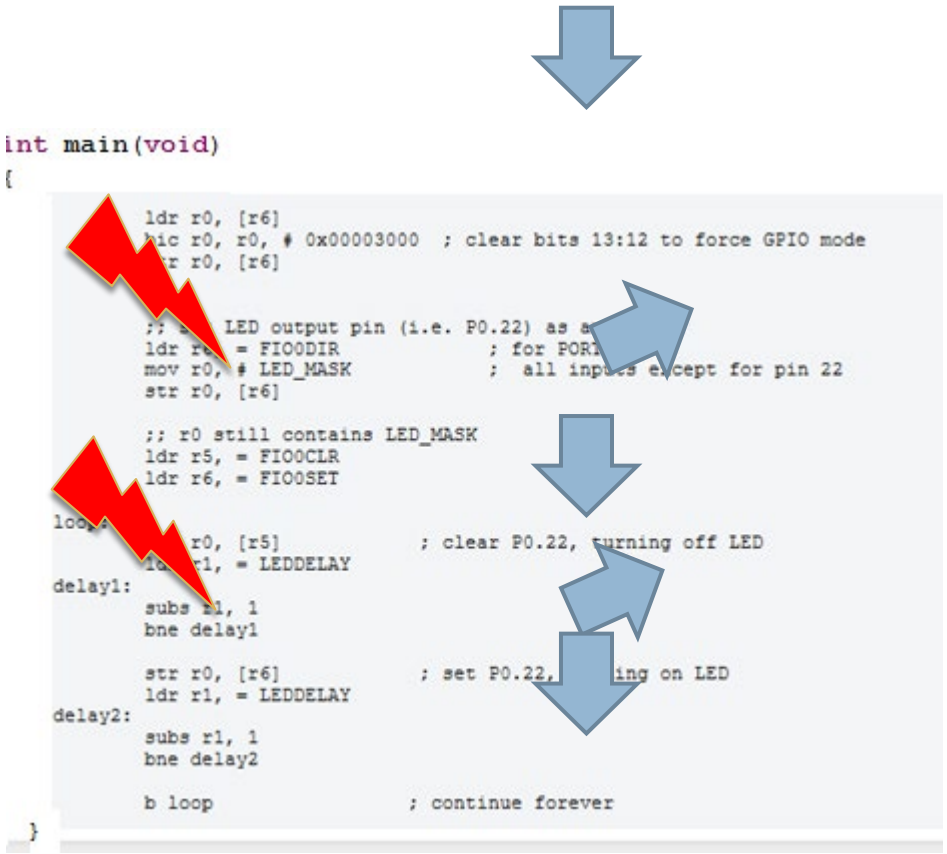


La atención de cada periférico es:

¡ASINCRONICA CON LA APLICACION !!!!

Interrupciones

COMPILAMOS Y LINKEAMOS



```
int main(void)
{
        ldr r0, [r6]
        bic r0, r0, # 0x00003000  ; clear bits 13:12 to force GPIO mode
        str r0, [r6]

        ;; set LED output pin (i.e. P0.22) as a
        ldr r6, = FIO0DIR              ; for PORT
        mov r0, # LED_MASK             ;  all inputs except for pin 22
        str r0, [r6]

        ;; r0 still contains LED_MASK
        ldr r5, = FIO0CLR
        ldr r6, = FIO0SET

loop
        str r0, [r5]            ; clear P0.22, turning off LED
        ldr r1, = LEDDELAY
delay1:
        subs r1, 1
        bne delay1

        str r0, [r6]            ; set P0.22, turning on LED
        ldr r1, = LEDDELAY
delay2:
        subs r1, 1
        bne delay2

        b loop                  ; continue forever
}
```

```
void funcion ( void )
{
    static char cont;
    cont ++;
    cont %= 2;

    if ( cont )
        corrimiento = 5 ;
    else
        corrimiento = 3 ;
}
```

Tabla de vectores de interrupcion

- Si una interrupción está habilitada, debe existir un mecanismo que vincule esa interrupción con su bloque de código (ISR) asociado.
- Ese mecanismo se denomina TABLA DE VECTORES DE INTERRUPCION.
- La vinculación ES el Vector de Interrupción.

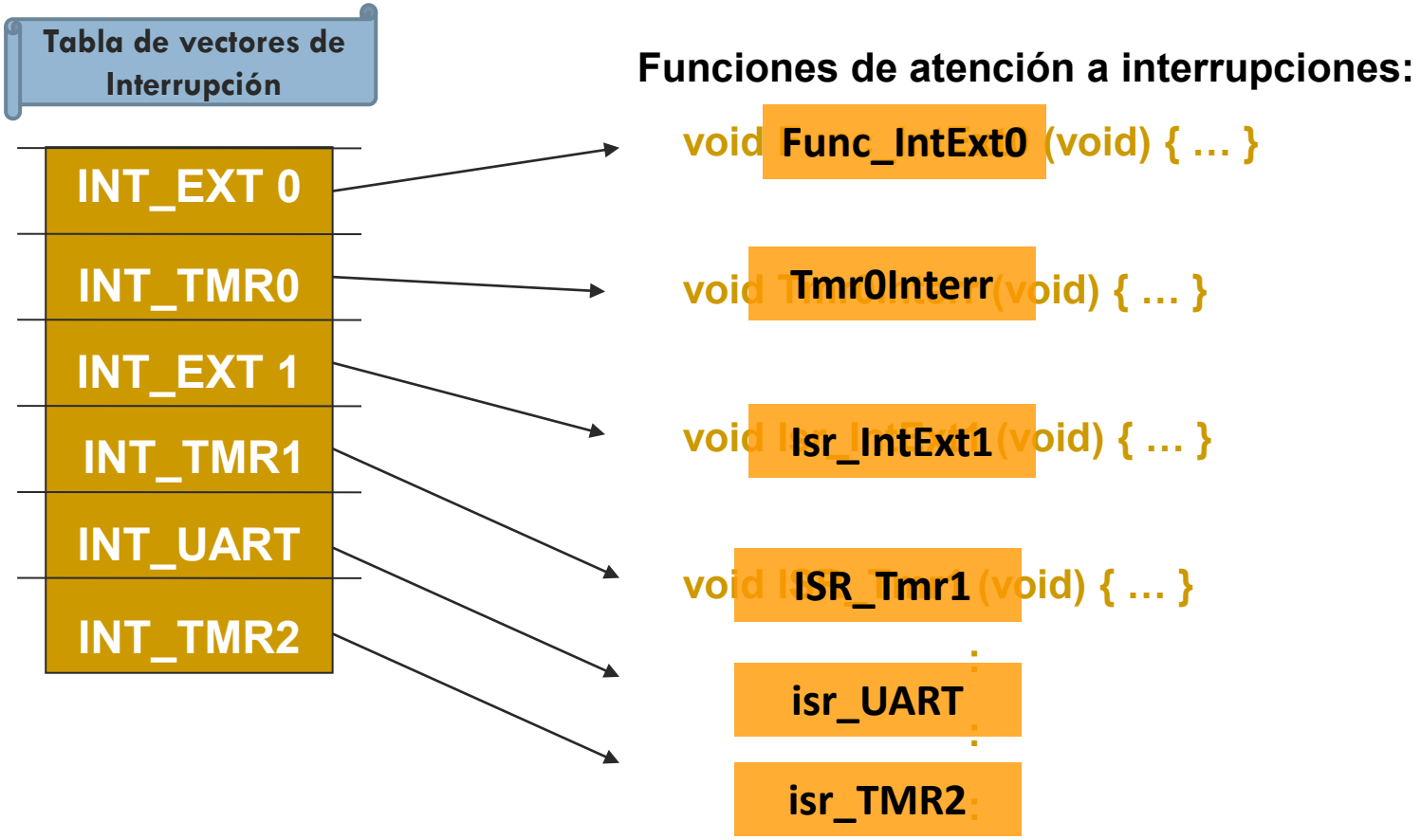


Tabla de vectores de interrupción

(core cortexM3)

#	priority	Type of Priorty	Acronym	Description	Address
	-	-	-	Reserved	0x0000_0000
1	-3	fixed	Reset	Reset	0x0000_0004
2	-2	fixed	UMI	Non maskable interrupt	0x0000_0008
3	-1	settable	HardFault	All class of fault	0x0000_000C
4	0	settable	MemManage	Memory management	0x0000_0010
5	1	settable	BusFault	Pre-fetch fault, memory access fault	0x0000_0014
6	2	settable	UsageFault	Undefined instruction or illegal state	0x0000_0018
7 a 10	-	-	-	Reserved	0x0000_001C 0x0000_002B
11	3	settable	SCCall	System service call via SWI instruction	0x0000_002C
12	4	settable	Debug Monitor	Debug Monitor	0x0000_0030
13	-	-	-	Reserved	0x0000_0034
14	5	settable	PendSV	Pendable request for system service	0x0000_0038
15	6	Settable	SysTick	System tick timer	0x0000_003C

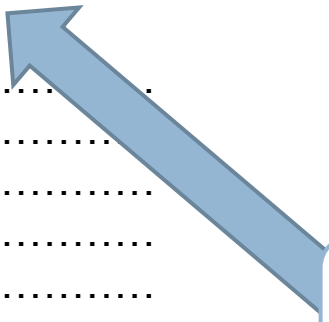
Tabla de vectores de interrupción

Table 50. Connection of interrupt sources to the Vectored Interrupt Controller

Interrupt ID	Exception Number	Vector Offset	Function	Flag(s)
0	16	0x40	WDT	Watchdog Interrupt (WDINT)
1	17	0x44	Timer 0	Match 0 - 1 (MR0, MR1) Capture 0 - 1 (CR0, CR1)
2	18	0x48	Timer 1	Match 0 - 2 (MR0, MR1, MR2) Capture 0 - 1 (CR0, CR1)
3	19	0x4C	Timer 2	Match 0-3 Capture 0-1
⋮				
16	32	0x80	PLL0 (Main PLL)	PLL0 Lock (PLOCK0)
17	33	0x84	RTC	Counter Increment (RTCCIF) Alarm (RTCALF)
18	34	0x88	External Interrupt	External Interrupt 0 (EINT0)
19	35	0x8C	External Interrupt	External Interrupt 1 (EINT1)
20	36	0x90	External Interrupt	External Interrupt 2 (EINT2)
21	37	0x94	External Interrupt	External Interrupt 3 (EINT3). **Note:** EINT3 channel is shared with GPIO interrupts
22	38	0x98	ADC	A/D Converter end of conversion
⋮				
32	48	0xC0	PLL1 (USB PLL)	PLL1 Lock (PLOCK1)
33	49	0xC4	USB Activity Interrupt	USB_NEED_CLK
34	50	0xC8	CAN Activity Interrupt	CAN1WAKE, CAN2WAKE

Funciones

```
void Funcion(void)
{
    ……
    ……
    ……
    ……
    ……
    ……
    ……
    ……
}
```

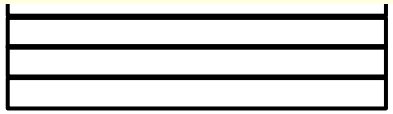


Recordando…

Representa la dirección de comienzo

Tabla de vectores de interrupción

```
//*****************************************************************************
Forward declaration of the default handlers. These are aliased.
When the application defines a handler (with the same name),
this will automatically take precedence over these weak
definitions
*****************************************************************************/
    void ResetISR(void);
WEAK void NMI_Handler(void);
WEAK void HardFault_Handler(void);
WEAK void MemManage_Handler(void);
WEAK void BusFault_Handler(void);
WEAK void UsageFault_Handler(void);
WEAK void SVCall_Handler(void);
WEAK void DebugMon_Handler(void);
WEAK void PendSV_Handler(void);
WEAK void SysTick_Handler(void);
WEAK void IntDefaultHandler(void);
```



¿Qué es el modificador WEAK?

FW_Drivers

main.c

```
WEAK void funcion (void);
void funcion ( void )
{

    printf( "Hola Mundo");


}
```

Débil

Integrador.c

```
void funcion ( void )
{

    printf( "Chau Mundo");


}
```

Fuerte

Toma el control

¿Cómo está usado aquí el modificador WEAK?

```
    void ResetISR(void);
WEAK void NMI_Handler(void);
WEAK void HardFault_Handler(void);
WEAK void MemManage_Handler(void);
WEAK void BusFault_Handler(void);
WEAK void UsageFault_Handler(void);
WEAK void SVCall_Handler(void);
WEAK void DebugMon_Handler(void);
WEAK void PendSV_Handler(void);
WEAK void SysTick_Handler(void);
WEAK void IntDefaultHandler(void);
```

?

```
void HardFault_Handler(void)
{
    while(1)
    {
    }
}
```

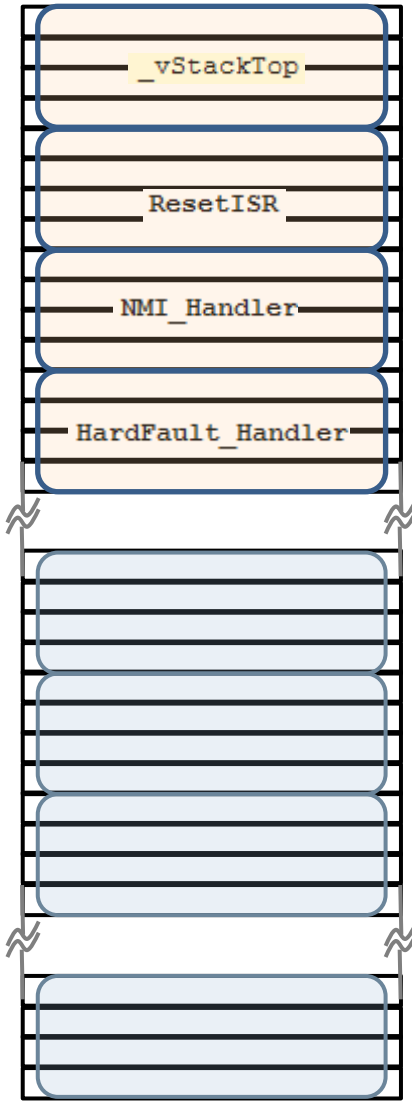
```
void TIMER0_IRQHandler(void) ALIAS(IntDefaultHandler);
void TIMER1_IRQHandler(void) ALIAS(IntDefaultHandler);
void TIMER2_IRQHandler(void) ALIAS(IntDefaultHandler);
void TIMER3_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART0_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART1_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART2_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART3_IRQHandler(void) ALIAS(IntDefaultHandler);
void PWM1_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C0_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C1_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C2_IRQHandler(void) ALIAS(IntDefaultHandler);
void SPI_IRQHandler(void) ALIAS(IntDefaultHandler);
void SSP0_IRQHandler(void) ALIAS(IntDefaultHandler);
void SSP1_IRQHandler(void) ALIAS(IntDefaultHandler);
void PLL0_IRQHandler(void) ALIAS(IntDefaultHandler);
void RTC_IRQHandler(void) ALIAS(IntDefaultHandler);
void EINT0_IRQHandler(void) ALIAS(IntDefaultHandler);
void EINT1_IRQHandler(void) ALIAS(IntDefaultHandler);
void EINT2_IRQHandler(void) ALIAS(IntDefaultHandler);
void EINT3_IRQHandler(void) ALIAS(IntDefaultHandler);
void ADC_IRQHandler(void) ALIAS(IntDefaultHandler);
void BOD_IRQHandler(void) ALIAS(IntDefaultHandler);
void USB_IRQHandler(void) ALIAS(IntDefaultHandler);
void CAN_IRQHandler(void) ALIAS(IntDefaultHandler);
void DMA_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2S_IRQHandler(void) ALIAS(IntDefaultHandler);
void ENET_IRQHandler(void) ALIAS(IntDefaultHandler);
void RIT_IRQHandler(void) ALIAS(IntDefaultHandler);
void MCPWM_IRQHandler(void) ALIAS(IntDefaultHandler);
void QEI_IRQHandler(void) ALIAS(IntDefaultHandler);
void PLL1_IRQHandler(void) ALIAS(IntDefaultHandler);
void USBActivity_IRQHandler(void) ALIAS(IntDefaultHandler);
```

Lo vemos en unos minutos...

Terminamos de construir la **IVT**

g_pfnVectors →

Interrupciones/
excepciones
CORTEX M3
(hasta Systick)




```
void EINT2_IRQHandler ( void )
{

    EINT2_IRQHandler
}
```

Table 50. Connection of interrupt sources to the Vectored Interrupt Controller

Interrupt ID	Exception Number	Vector Offset	Function	Flag(s)
0	16	0x40	WDT	Watchdog Interrupt (WDINT)
1	17	0x44	Timer 0	Match 0 - 1 (MR0, MR1) Capture 0 - 1 (CR0, CR1)
2	18			
3	19			
⋮				
16	32	0x80	PLL0 (Main PLL)	PLL0 Lock (PLOCK0)
17	33			
18	34			
19	35			
20	36			
21	37	0x94	External Interrupt	External Interrupt 3 (EINT3). **Note:** EINT3 channel is shared with GPIO interrupts
22	38	0x98	ADC	A/D Converter end of conversion
⋮				
32	48	0xC0	PLL1 (USB PLL)	PLL1 Lock (PLOCK1)
33	49	0xC4	USB Activity Interrupt	USB_NEED_CLK
34	50	0xC8	CAN Activity Interrupt	CAN1WAKE, CAN2WAKE

```
void EINT3_IRQHandler ( void )
{

}
```

```
void ADC_IRQHandler ( void )
{

}
```

```
void CANActivity_IRQHandler ( void )
{

}
```

32	48	0xC0	PLL1 (USB PLL)	PLL1 Lock (PLOCK1)
33	49	0xC4	USB Activity Interrupt	USB_NEED_CLK
34	50	0xC8	CAN Activity Interrupt	CAN1WAKE, CAN2WAKE

19	35	0x8C	External Interrupt	External Interrupt 1 (EINT1)
20	36	0x90	External Interrupt	External Interrupt 2 (EINT2)
21	37	0x94	External Interrupt	External Interrupt 3 (EINT3). **Note:** EINT3 channel is shared w
22	38	0x98	ADC	A/D Converter end of conversion

Tabla de vectores de interrupción

Nombres que pone el compilador

Table 50. Connection of interrupt sources to the Vectored Interrupt Controller

Interrupt ID	Exception Number	Vector Offset	Function	Flag(s)
0	16	0x40	WDT	Watchdog Interrupt (WDINT)
1	17	0x44	Timer 0	Match 0 - 1 (MR0, MR1) Capture 0 - 1 (CR0, CR1)
2	18	0x48	Timer 1	Match 0 - 2 (MR0, MR1, MR2) Capture 0 - 1 (CR0, CR1)
3	19	0x4C	Timer 2	Match 0-3 Capture 0-1
⋮				
16	32	0x80	PLL0 (Main PLL)	PLL0 Lock (PLOCK0)
17	33	0x84	RTC	Counter Increment (RTCCIF) Alarm (RTCALF)
18	34	0x88	External Interrupt	External Interrupt 0 (EINT0)
19	35	0x8C	External Interrupt	External Interrupt 1 (EINT1)
20	36	0x90	External Interrupt	External Interrupt 2 (EINT2)
21	37	0x94	External Interrupt	External Interrupt 3 (EINT3). **Note:** EINT3 channel is shared with GP
22	38	0x98	ADC	A/D Converter end of conversion
⋮				
32	48	0xC0	PLL1 (USB PLL)	PLL1 Lock (PLOCK1)
33	49	0xC4	USB Activity Interrupt	USB_NEED_CLK
34	50	0xC8	CAN Activity Interrupt	CAN1WAKE, CAN2WAKE

```
WDT_IRQHandler
TIMER0_IRQHandler
TIMER1_IRQHandler
TIMER2_IRQHandler
PLL0_IRQHandler
RTC_IRQHandler
EINT0_IRQHandler
EINT1_IRQHandler
EINT2_IRQHandler
EINT3_IRQHandler
ADC_IRQHandler
PLL1_IRQHandler
USBActivity_IRQHandler
CANActivity_IRQHandler
```

ALIAS

¿Cómo está usado aquí el modificador WEAK?

```
    void ResetISR(void);
WEAK void NMI_Handler(void);
WEAK void HardFault_Handler(void);
WEAK void MemManage_Handler(void);
WEAK void BusFault_Handler(void);
WEAK void UsageFault_Handler(void);
WEAK void SVCall_Handler(void);
WEAK void DebugMon_Handler(void);
WEAK void PendSV_Handler(void);
WEAK void SysTick_Handler(void);
WEAK void IntDefaultHandler(void);
```

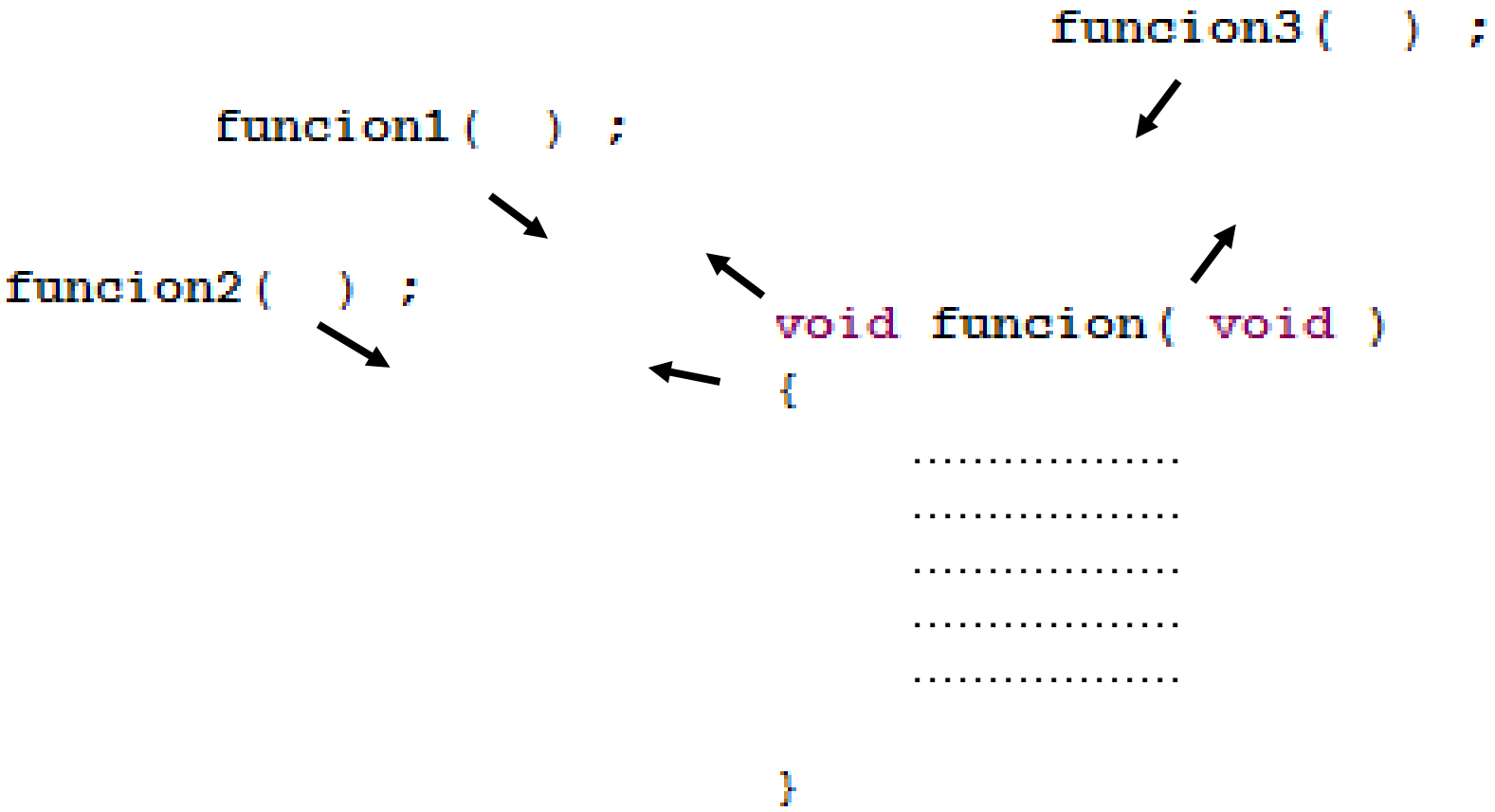
?

```
void TIMER0_IRQHandler(void) ALIAS(IntDefaultHandler);
void TIMER1_IRQHandler(void) ALIAS(IntDefaultHandler);
void TIMER2_IRQHandler(void) ALIAS(IntDefaultHandler);
void TIMER3_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART0_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART1_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART2_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART3_IRQHandler(void) ALIAS(IntDefaultHandler);
void PWM1_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C0_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C1_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C2_IRQHandler(void) ALIAS(IntDefaultHandler);
void SPI_IRQHandler(void) ALIAS(IntDefaultHandler);
void SSP0_IRQHandler(void) ALIAS(IntDefaultHandler);
void SSP1_IRQHandler(void) ALIAS(IntDefaultHandler);
void PLL0_IRQHandler(void) ALIAS(IntDefaultHandler);
void RTC_IRQHandler(void) ALIAS(IntDefaultHandler);
void EINT0_IRQHandler(void) ALIAS(IntDefaultHandler);
void EINT1_IRQHandler(void) ALIAS(IntDefaultHandler);
void EINT2_IRQHandler(void) ALIAS(IntDefaultHandler);
void EINT3_IRQHandler(void) ALIAS(IntDefaultHandler);
void ADC_IRQHandler(void) ALIAS(IntDefaultHandler);
void BOD_IRQHandler(void) ALIAS(IntDefaultHandler);
void USB_IRQHandler(void) ALIAS(IntDefaultHandler);
void CAN_IRQHandler(void) ALIAS(IntDefaultHandler);
void DMA_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2S_IRQHandler(void) ALIAS(IntDefaultHandler);
void ENET_IRQHandler(void) ALIAS(IntDefaultHandler);
void RIT_IRQHandler(void) ALIAS(IntDefaultHandler);
void MCPWM_IRQHandler(void) ALIAS(IntDefaultHandler);
void QEI_IRQHandler(void) ALIAS(IntDefaultHandler);
void PLL1_IRQHandler(void) ALIAS(IntDefaultHandler);
void USBActivity_IRQHandler(void) ALIAS(IntDefaultHandler);
```

Lo vemos en unos minutos...

¿Qué es el modificador ALIAS?

```
void funcion1( void ) ALIAS( funcion );
void funcion2( void ) ALIAS( funcion );
void funcion3( void ) ALIAS( funcion );
```



¿Cómo está usado aquí el modificador ALIAS?

(ver **cr_startup_lpc176x.c**)

```
void TIMER1_IRQHandler(void) ALIAS(IntDefaultHandler);
void TIMER2_IRQHandler(void) ALIAS(IntDefaultHandler);
void TIMER3_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART0_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART1_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART2_IRQHandler(void) ALIAS(IntDefaultHandler);
void UART3_IRQHandler(void) ALIAS(IntDefaultHandler);
void PWM1_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C0_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C1_IRQHandler(void) ALIAS(IntDefaultHandler);
void I2C2_IRQHandler(void) ALIAS(IntDefaultHandler);
void SPI_IRQHandler(void) ALIAS(IntDefaultHandler);
void SSP0_IRQHandler(void) ALIAS(IntDefaultHandler);
void SSP1_IRQHandler(void) ALIAS(IntDefaultHandler);
void PLL0_IRQ
void RTC_IRQH
void EINT0_IR
void EINT1_IR
void EINT2_IR
void EINT3_IR
void ADC_IRQH
void BOD_IRQH
void USB_IRQH
void CAN_IRQH
void DMA_IRQH
void I2S_IRQHandler(void) ALIAS(IntDefaultHandler);
void ENET_IRQHandler(void) ALIAS(
void RIT_IRQHandler(void) ALIAS(I
void MCPWM_IRQHandler(void) ALIAS
void QEI_IRQHandler(void) ALIAS(I
void PLL1_IRQHandler(void) ALIAS(
void USBActivity_IRQHandler(void)
```

```
void IntDefaultHandler(void)
{
    while(1)
    {
    }
}
```

Si no defino una función que atienda a esta interrupción (como a cualquier otra)…. entonces será llamada **IntDefaultHandler()** *por defecto.*

Agradecimientos

Presentación basada en el trabajo del Ing. Marcelo Trujillo

Profesor Asociado – Informática II