Ingeniería de Sistemas Electrónicos

Programación avanzada de microcontroladores (II)

- Integración de referencia temporal en el sistema
- Desarrollo de aplicaciones “confiables”
- Reducción del consumo
- Desarrollo de software. Control de versiones
- Prácticas de laboratorio

Ingeniería de Sistemas Electrónicos

Programación avanzada de microcontroladores (II)

- Integración de referencia temporal en el sistema
- Desarrollo de aplicaciones “confiables”
- Reducción del consumo
- Desarrollo de software. Control de versiones
- Prácticas de laboratorio

Integración de referencia temporal en el sistema

As life becomes digitised and automated, exact time is increasingly important:

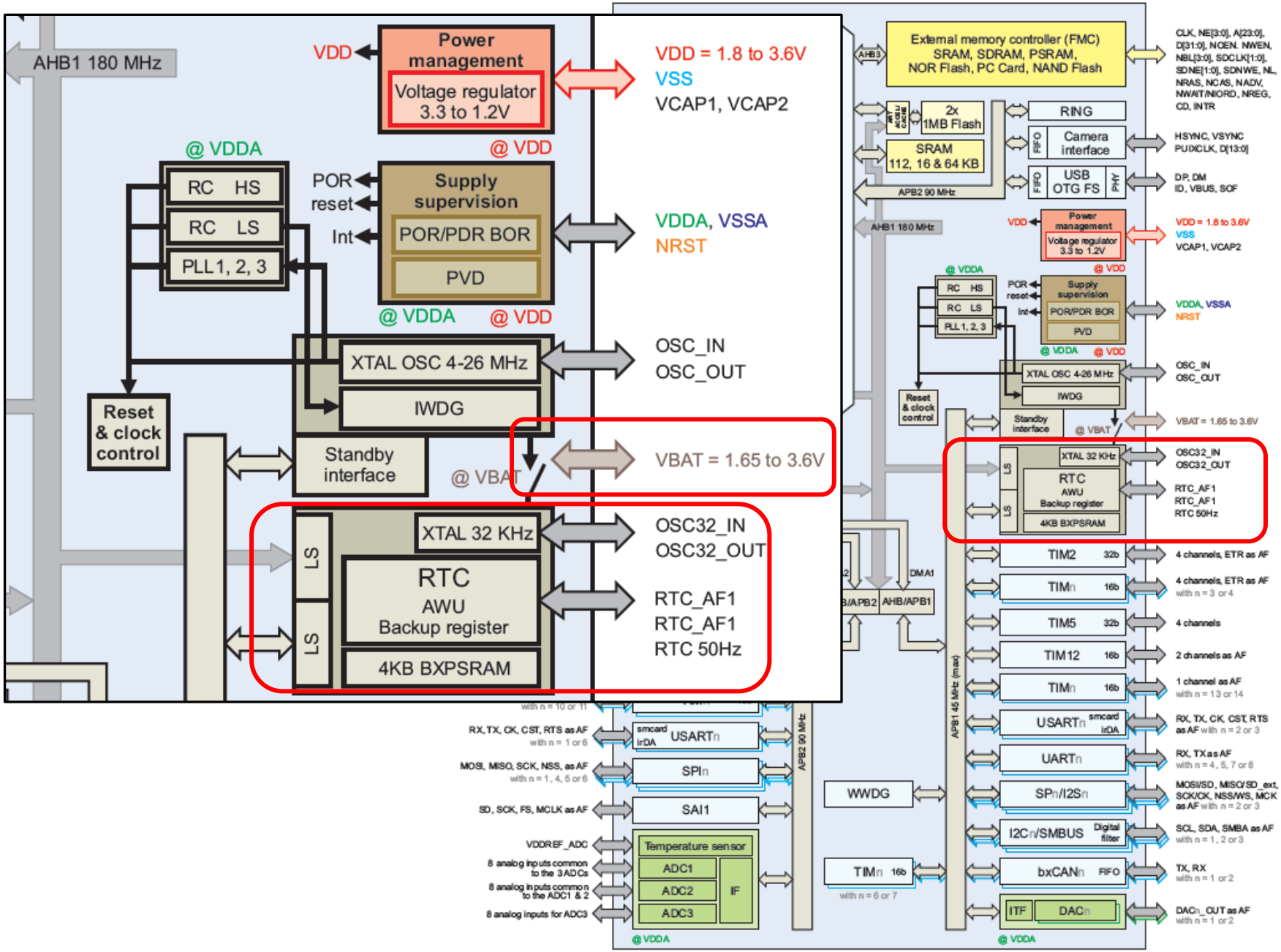
- The telecommunications industry relies on accurate time for the transfer of vast amounts of data.
- Utility companies use time synchronisation to manage power distribution.
- Financial services need exact time to timestamp transactions and ensure traceable records.
- Satellite navigation depends on precise time, with a difference of one microsecond causing a positioning error of 300 metres.[1]
- CCTV and speed cameras require an accurate timestamp to be admissible as evidence.
- Countless businesses rely on precise time to manage their day-to-day processes, such as synchronising clocking-in systems.

Timestamp

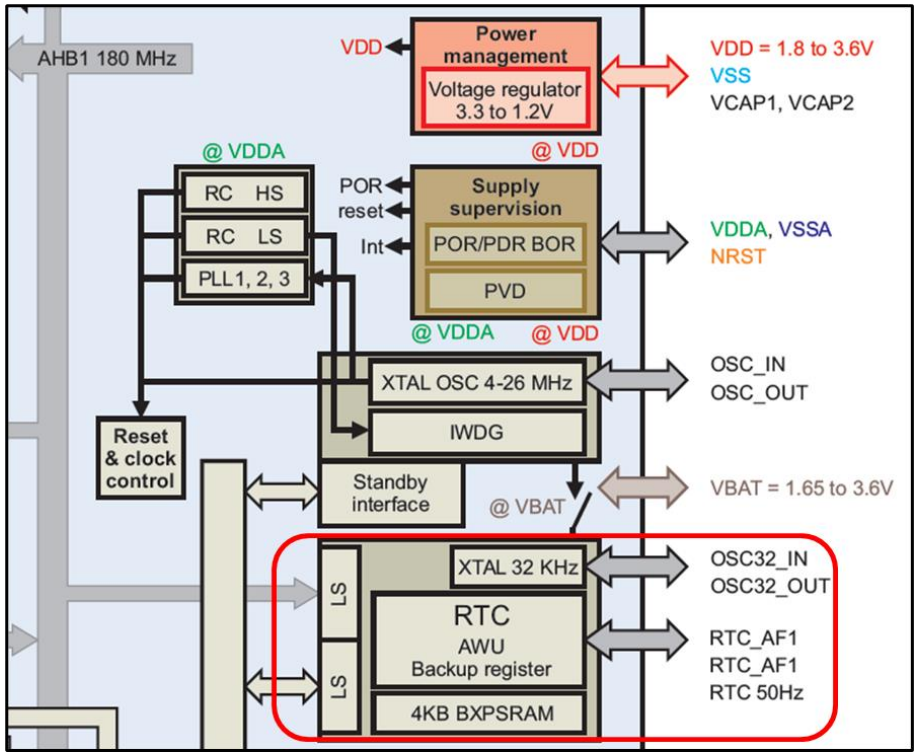
Formas de mantener una referencia temporal

- Reloj software "local"
  - Timer con interrupciones periódicas (cada segundo?)
  - Puesta en hora inicial
  - Mantenimiento de la hora
  - Desventajas??
- Reloj hardware (RTC, real time clock)
  - Hardware específico (interno o externo)
  - Puesta en hora
  - Mantenimiento de la hora automático
  - Desventajas??
- Servidor de tiempo en red
  - Proporciona la fecha y hora actuales
  - Protocolo SNTP (Simple Network Time Protocol), versión simplificada de NTP
  - Desventajas??

STM32F4 Real Time Clock (RTC)



STM32F4 Real Time Clock (RTC)



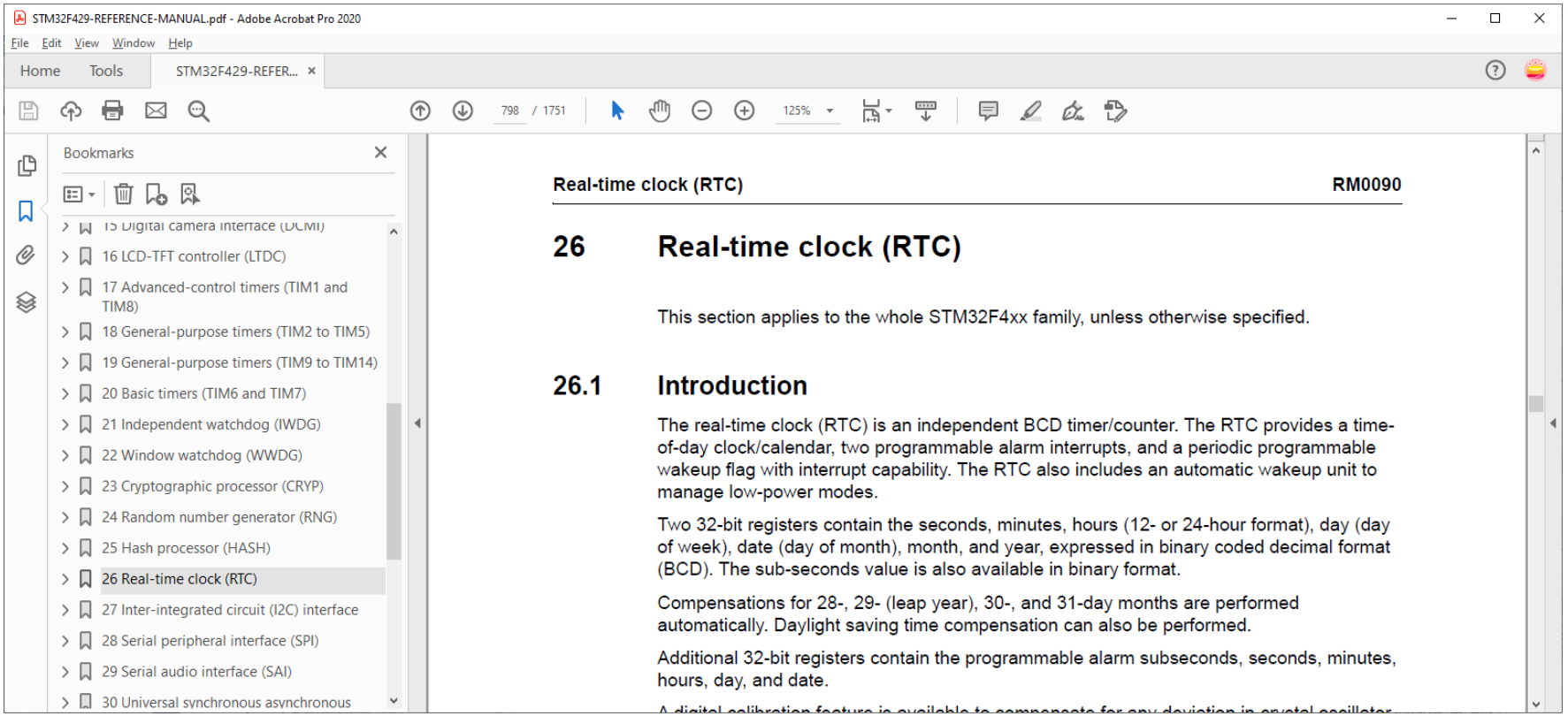
RTC:

- ✓ Special peripheral
- ✓ Independent power domain
- ✓ Independent oscillator
- ✓ Special timer with dedicated registers to store calendar and time data such as years, months, days, hours, minutes, and seconds.
- ✓ RTC plays a major role in power-saving strategies as it can be the only domain to stay awake while every other peripheral has been turned into standby or sleeping modes.

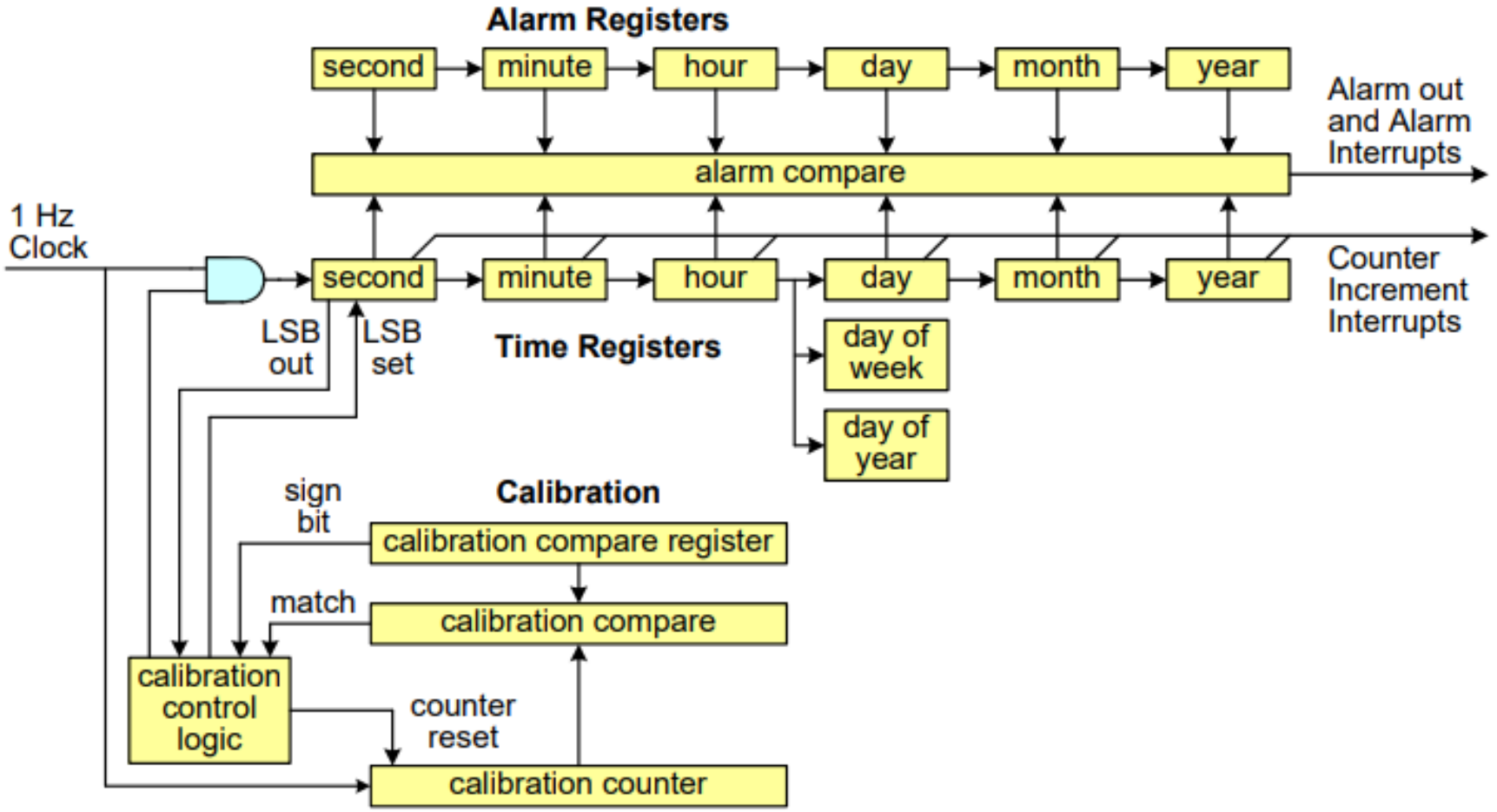
- ✓ Very low power consumption
- ✓ You can even turn the STM32 device off VDD and still keep the RTC running as long as you power supply the VBAT pin (using a button battery for instance).

STM32F4 Real Time Clock (RTC)

- ✓ All STM32 microcontrollers provide an integrated RTC unit that is not limited to keeping track of the current date/time. In fact, RTC provides some additional and relevant features such as antitampering detection, generation of alarm events and the ability to wake-up the MCU from deeper low-power modes.

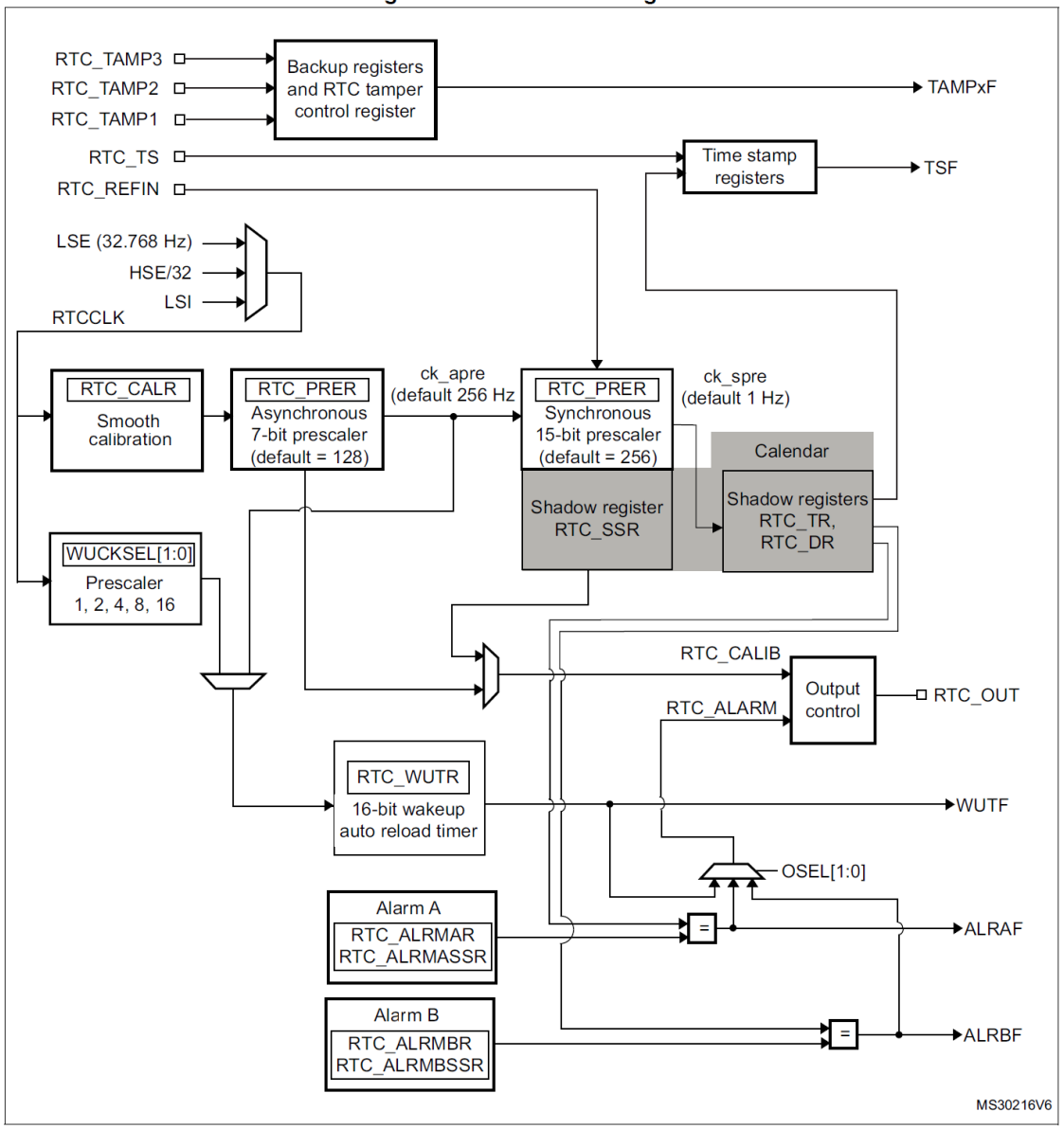


NXP LPC1768 (Cortex M3) Real Time Clock (RTC)



Real Time Clock (RTC) STM32F429

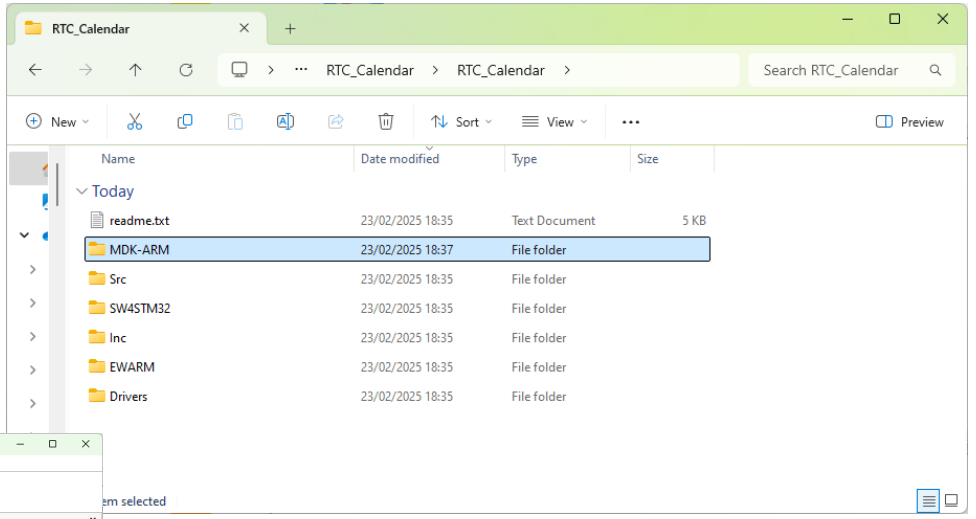
Figure 237. RTC block diagram

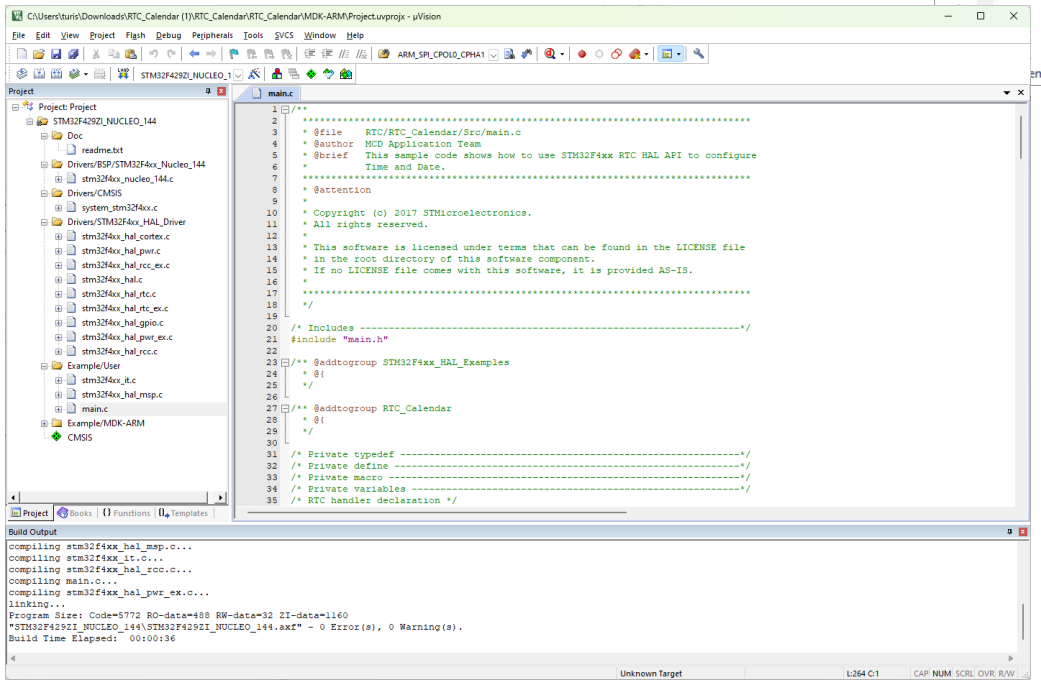


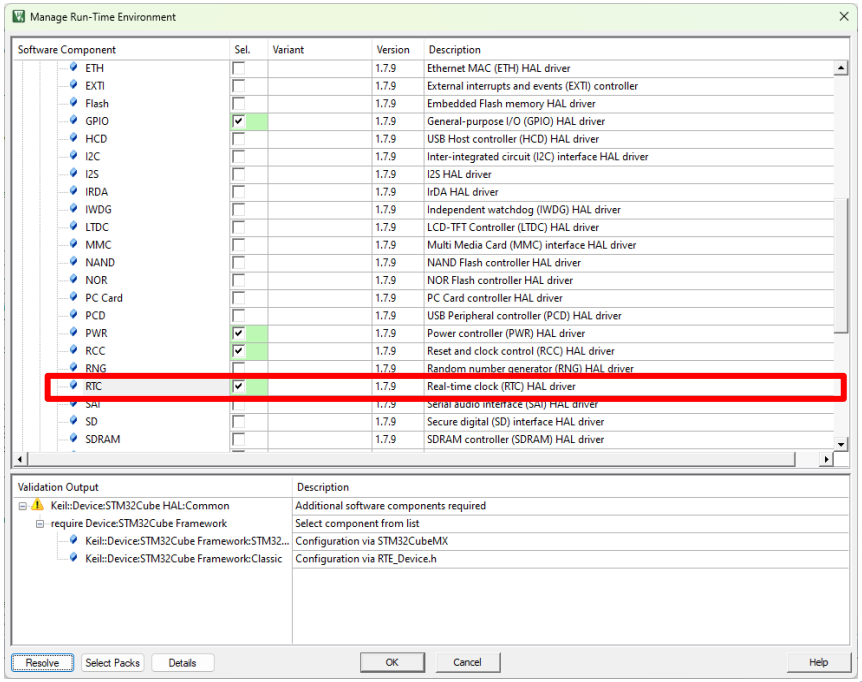
1. On STM32F4xx devices, the RTC_AF1 and RTC_AF2 alternate functions are connected to PC13 and PI8, respectively.

STM32F4 Real Time Clock (RTC)

Ejemplo RTC_Calendar (STM32CubeMX)





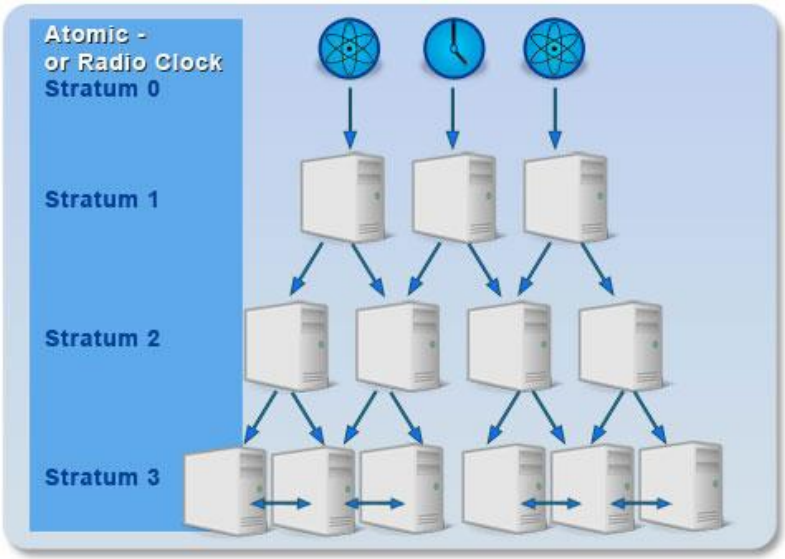


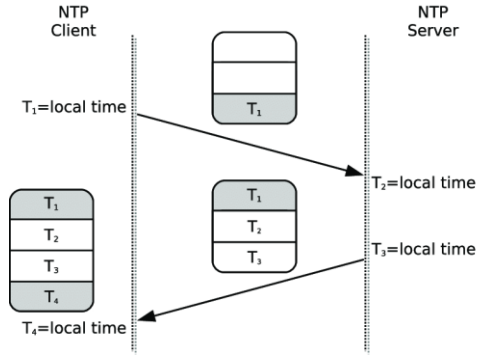
Formas de mantener una referencia temporal

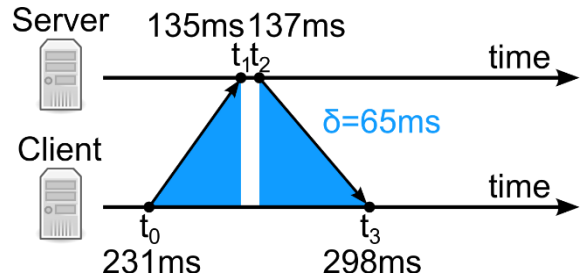
- Reloj software "local"
  - Timer con interrupciones periódicas (cada segundo?)
  - Puesta en hora inicial
  - Mantenimiento de la hora
  - Desventajas??
- Reloj hardware (RTC, real time clock)
  - Hardware específico (interno o externo)
  - Puesta en hora
  - Mantenimiento de la hora automático
  - Desventajas??
- Servidor de tiempo en red
  - Proporciona la fecha y hora actuales
  - Protocolo SNTP (Simple Network Time Protocol), versión simplificada de NTP
  - Desventajas??

Servidor de tiempo en red. Protocolos NTP y SNTP

Network Time Protocol (NTP) es un protocolo de Internet para sincronizar los relojes de los sistemas informáticos a través del enrutamiento de paquetes en redes con latencia variable. NTP utiliza UDP como su capa de transporte, usando el puerto 123. Está diseñado para resistir los efectos de la latencia variable.

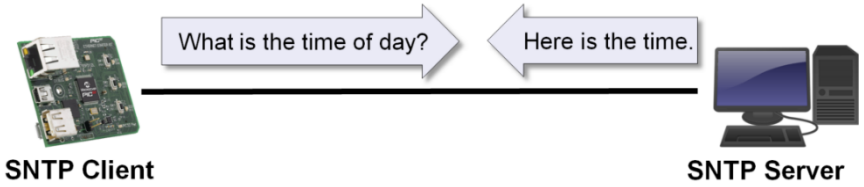




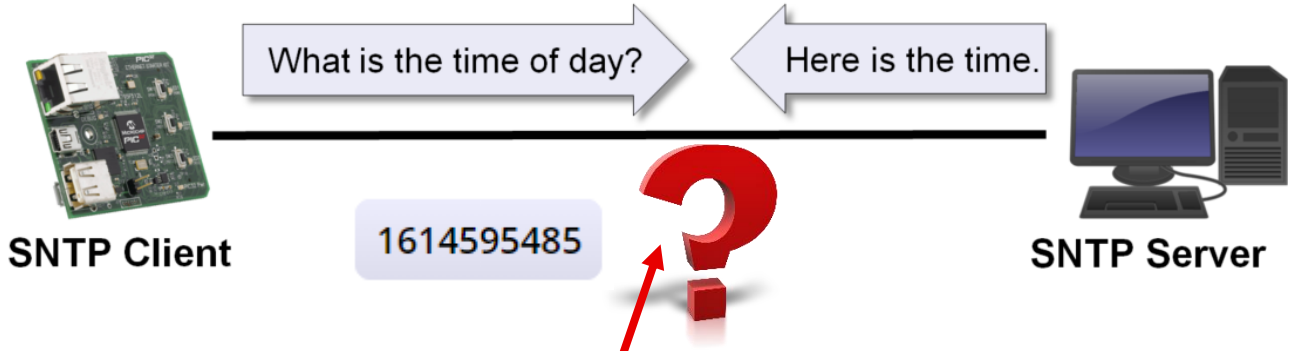


offset = $[(T_2 - T_1) + (T_3 - T_4)] / 2$,
delay = $(T_4 - T_1) - (T_3 - T_2)$.

Simple Network Time Protocol (**SNTP**) es una versión simplificada de NTP para sistemas con menor capacidad de procesamiento y menores requerimientos de tiempo. Es compatible con NTP, los paquetes intercambiados entre el cliente y el servidor de tiempo son los mismos, pero el cliente no necesita ejecutar algoritmos para determinar y mantener la sincronización.



Protocolo SNTP



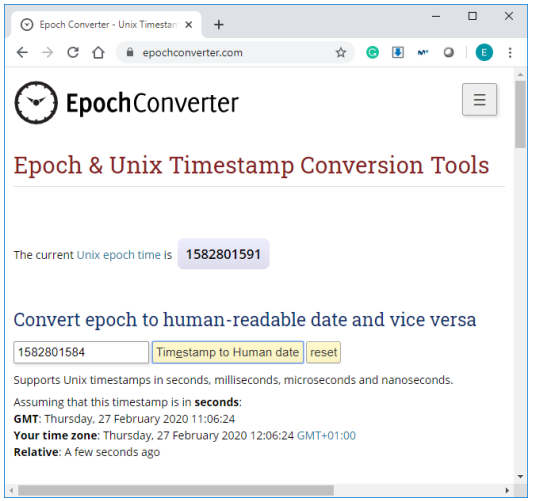
Network Time Foundation http://www.ntp.org/

NTP Pool Project

https://www.ntppool.org/ → Lista de servidores NTP

The current Unix epoch time is 1582801685

Segundos transcurridos desde el 1 de enero de 1970

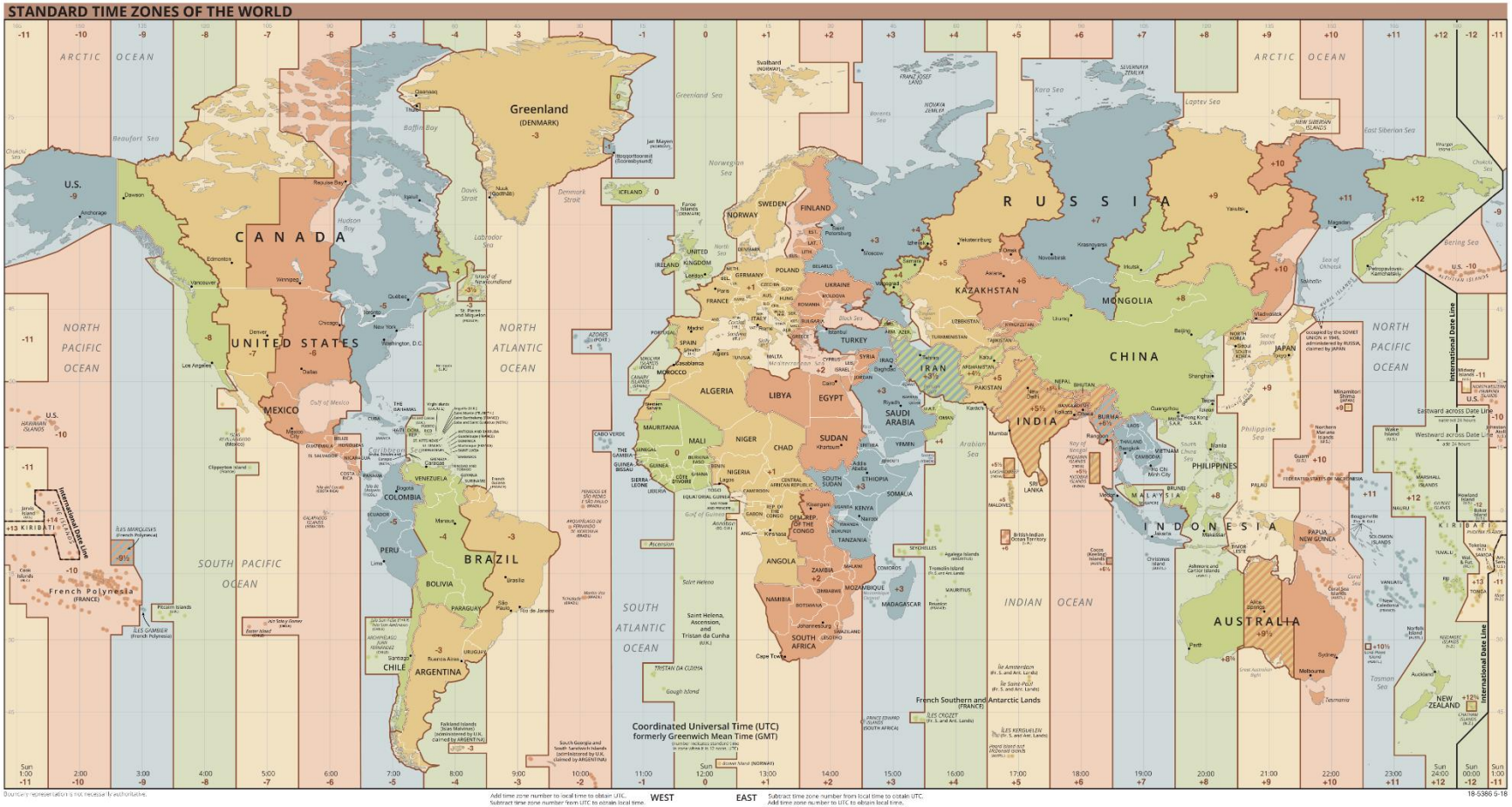


https://www.epochconverter.com/

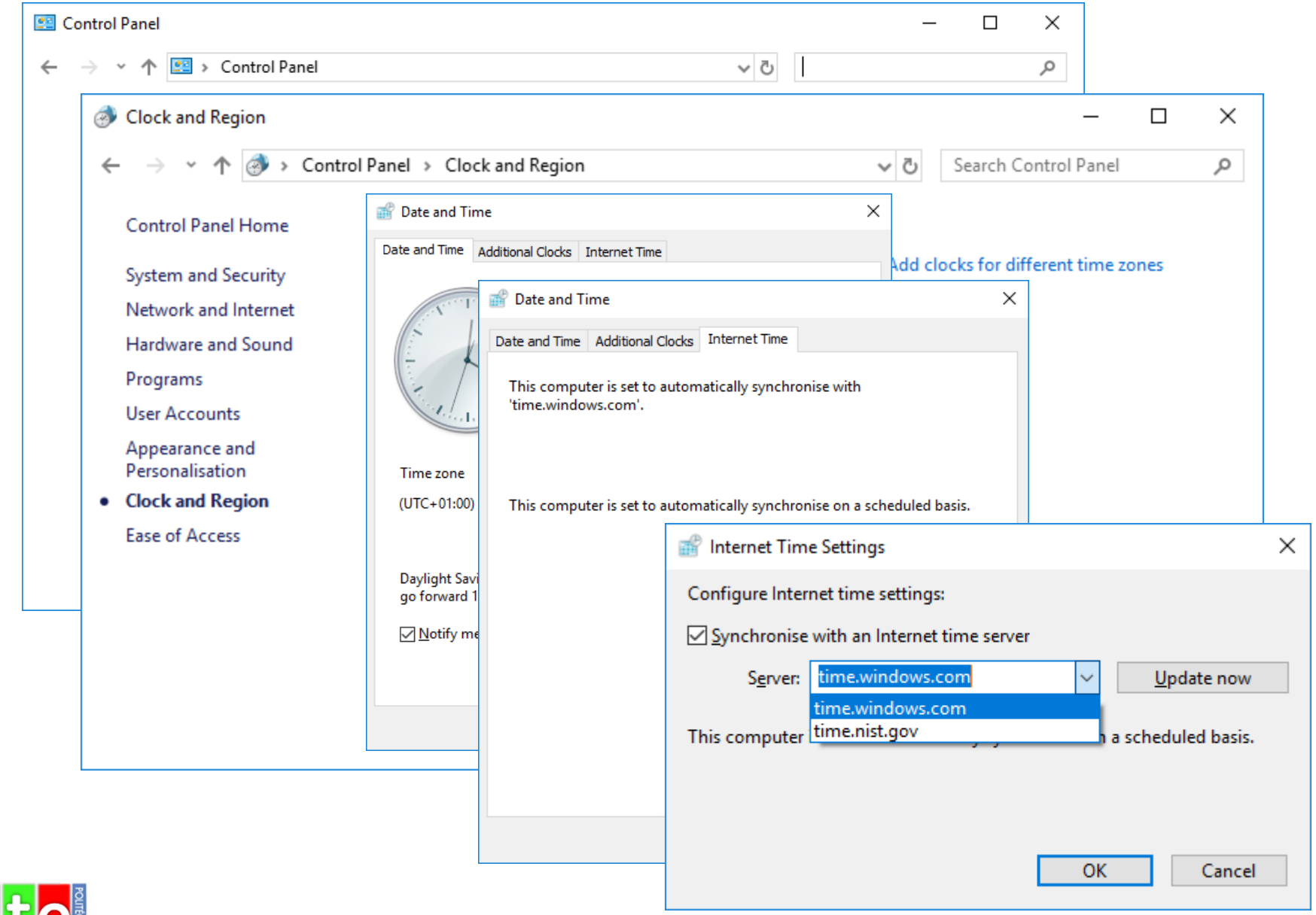
The current Unix epoch time is **1582801685**

Segundos transcurridos desde el 1 de enero de 1970

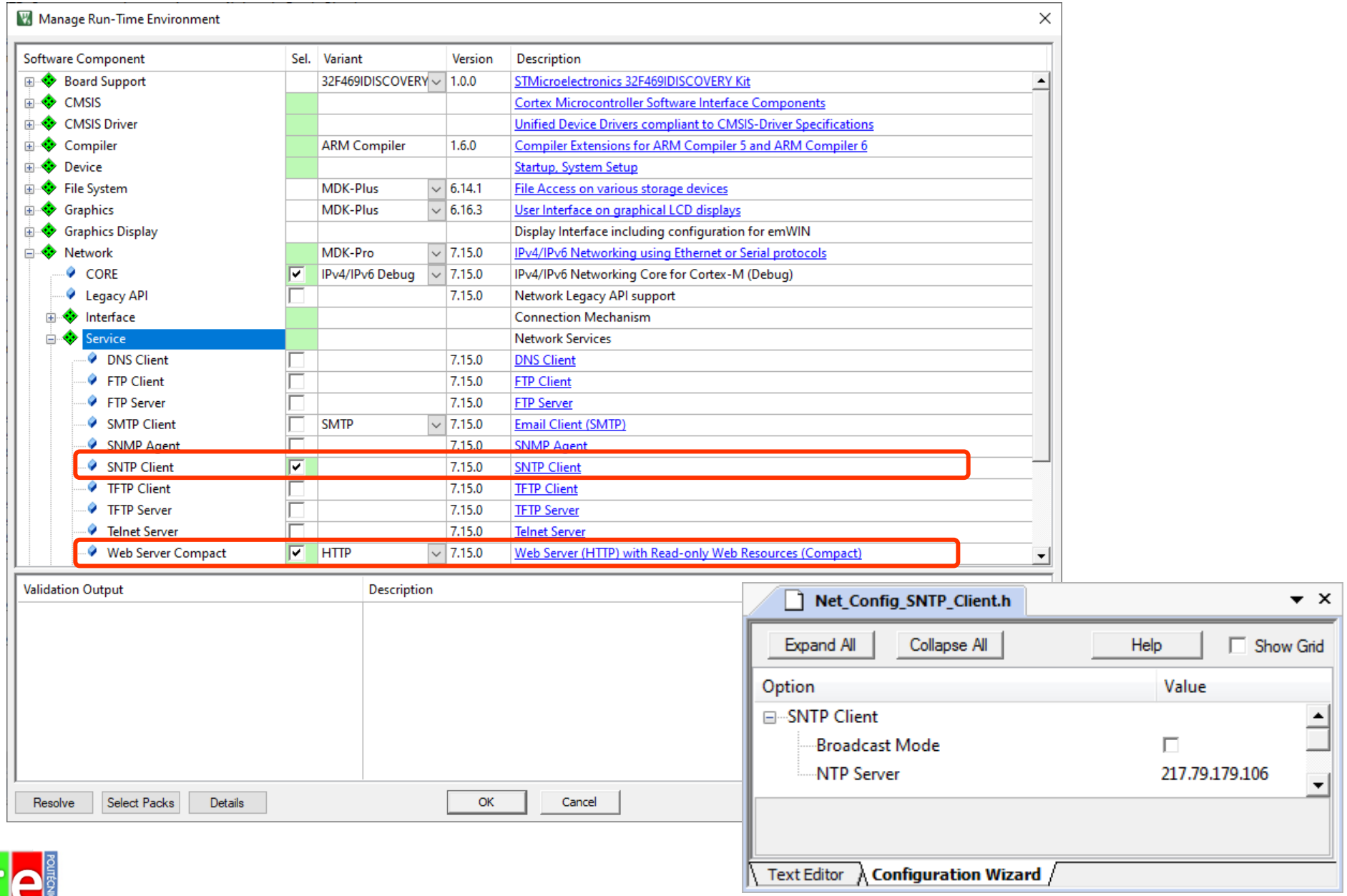
Hora UTC (Coordinated Universal Time)



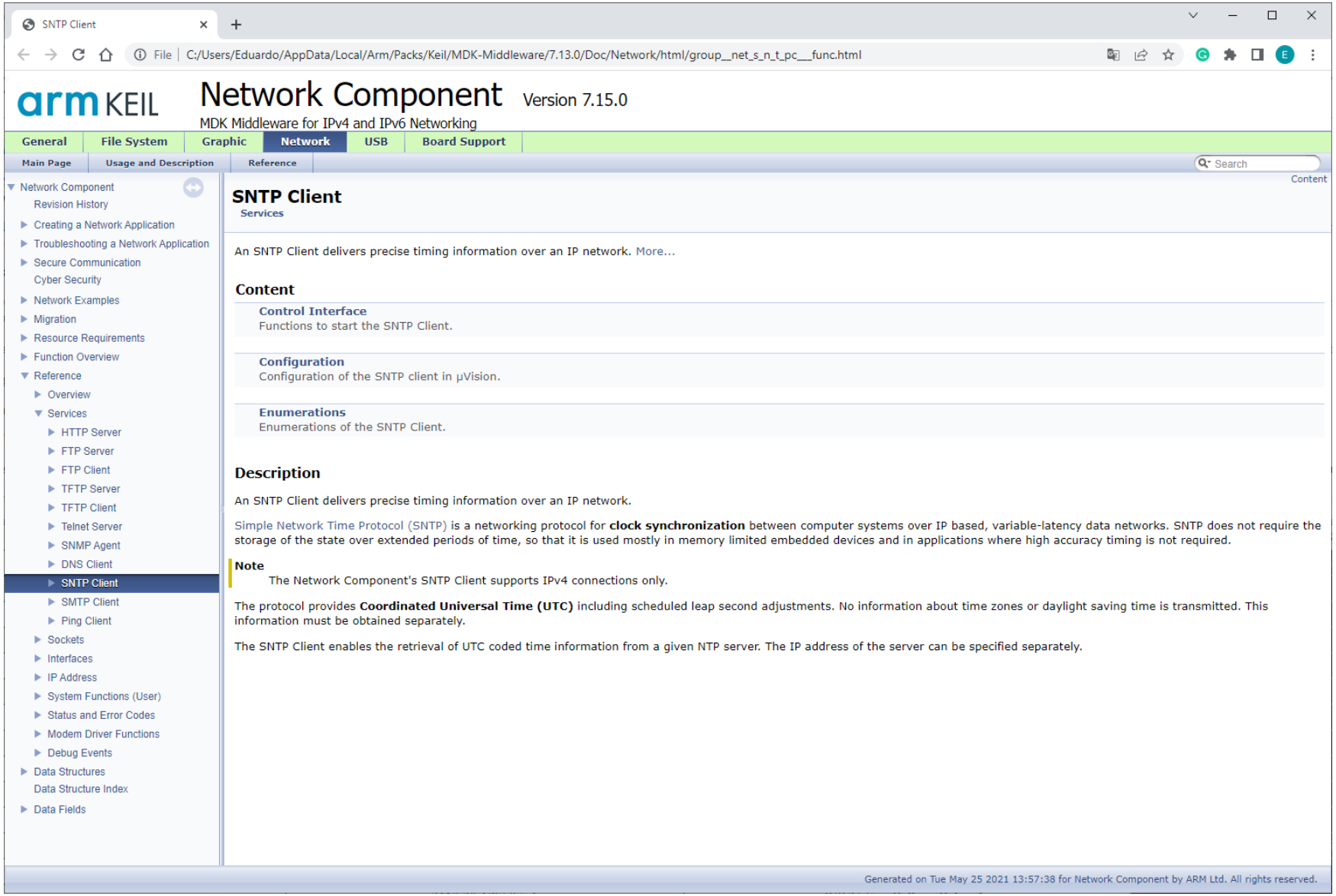
Protocolo SNTP. Uso en ordenadores, móviles…



Protocolo SNTP. Integración en aplicación servidor Web

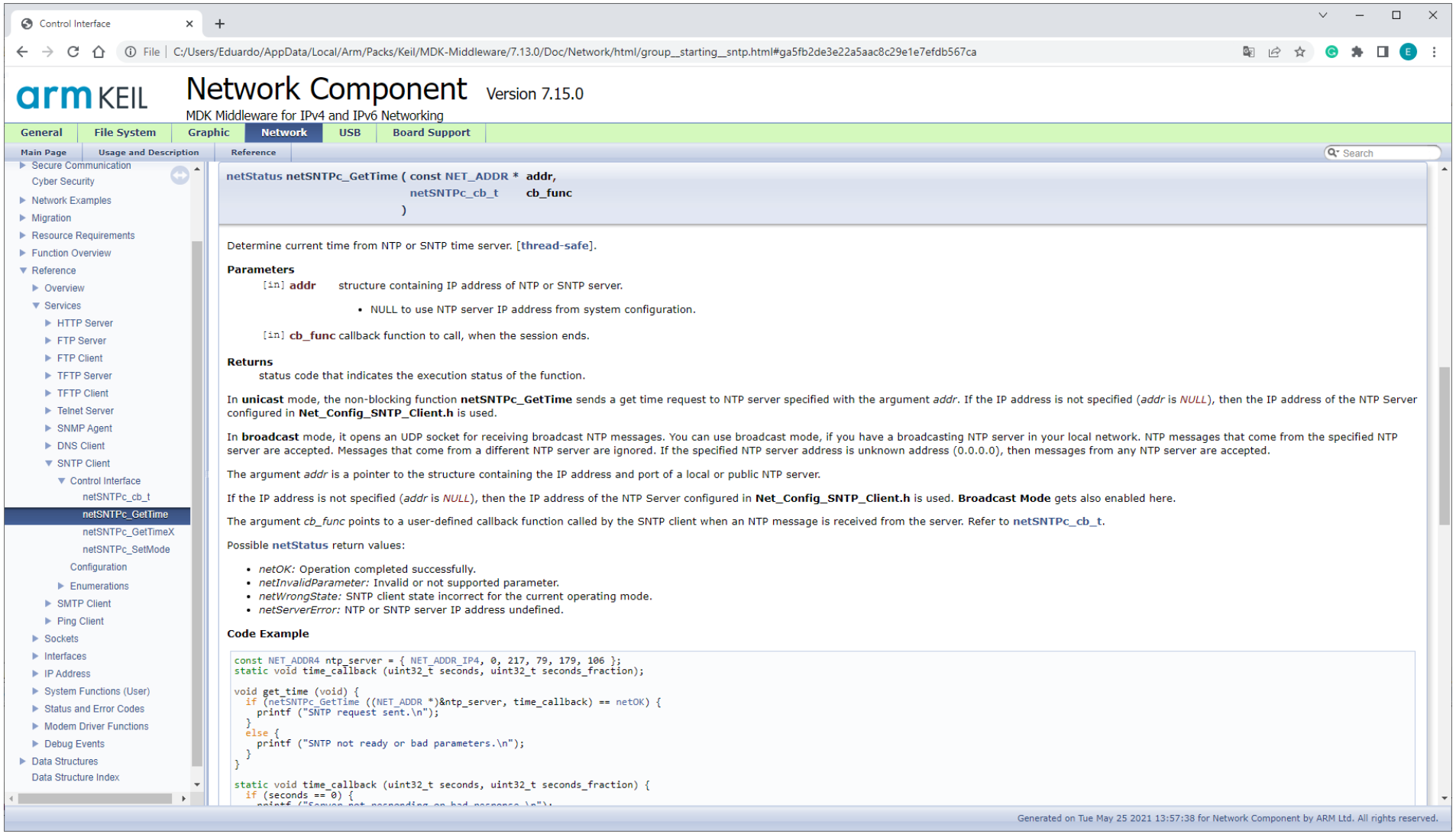


Protocolo SNTP. Integración en aplicación



No utilizar la info de la web (última version), sino la local (version utilizada)

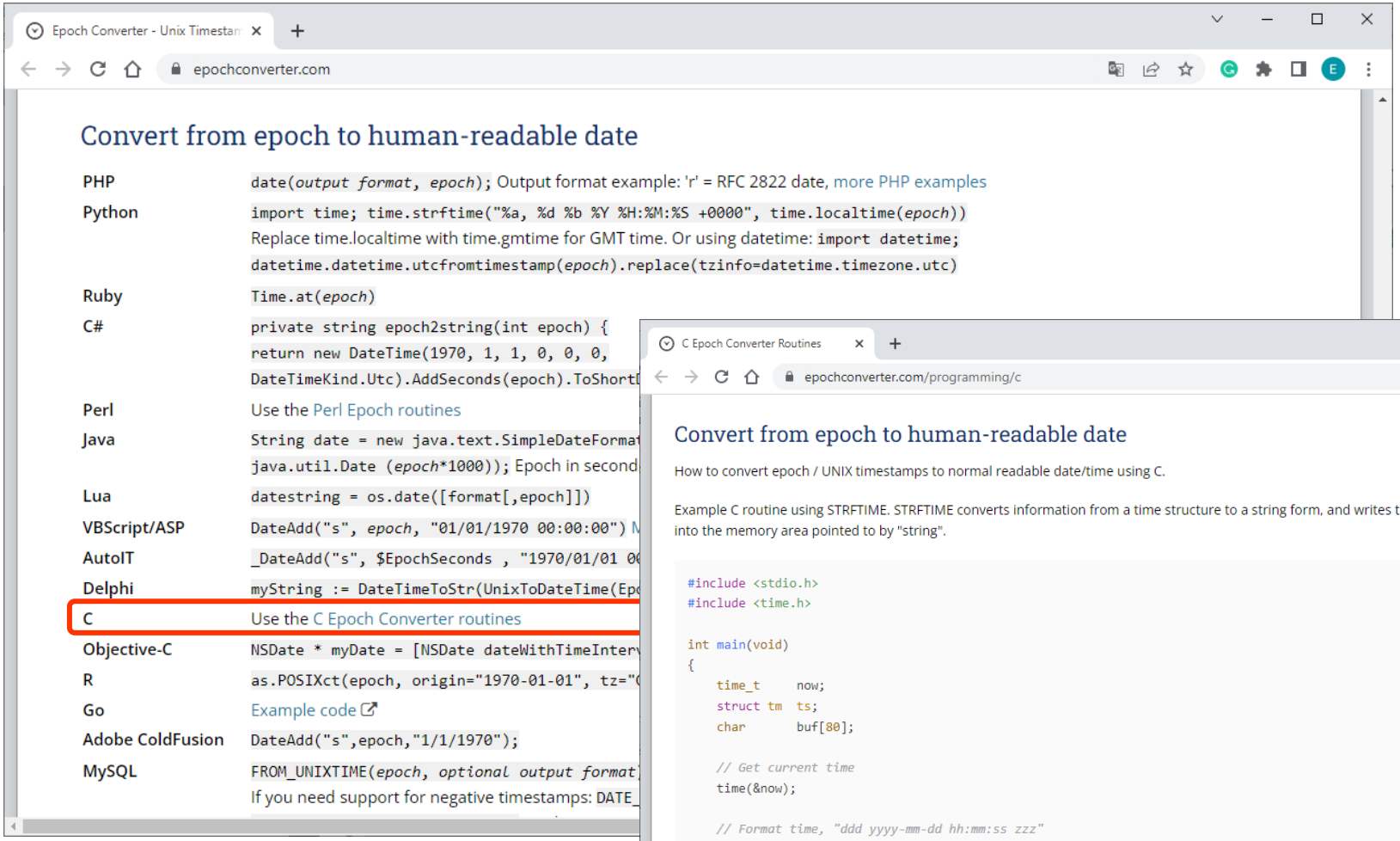
Protocolo SNTP. Integración en aplicación



Segundos desde 01/01/1970, utilizar función para su conversion.

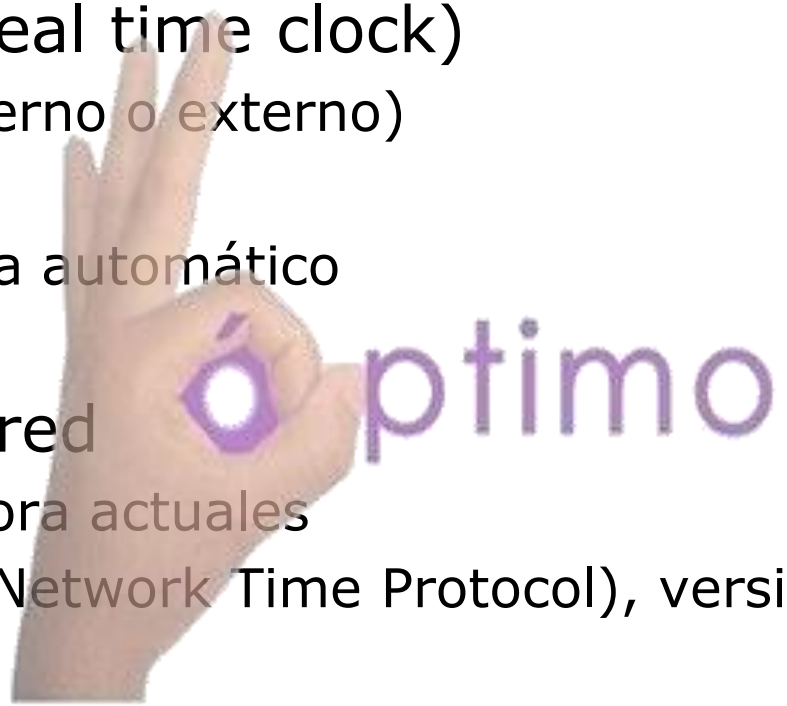
Protocolo SNTP. Integración en aplicación

https://www.epochconverter.com/programming/c

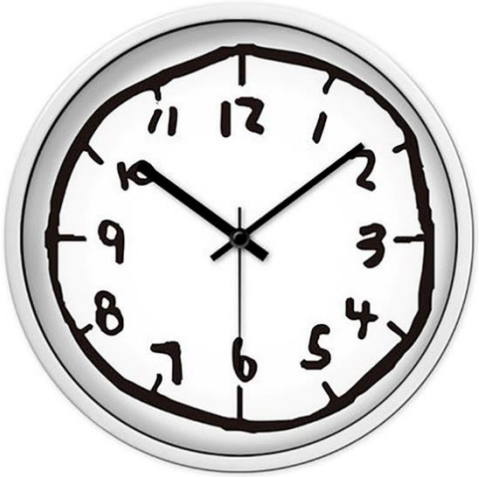


Formas de mantener una referencia temporal

- Reloj software "local"
  - Timer con interrupciones periódicas (cada segundo?)
  - Puesta en hora inicial
  - Mantenimiento de la hora
  - Desventajas??
- Reloj hardware (RTC, real time clock)
  - Hardware específico (interno o externo)
  - Puesta en hora
  - Mantenimiento de la hora automático
  - ~~Desventajas??~~
- Servidor de tiempo en red
  - Proporciona la fecha y hora actuales
  - Protocolo SNTP (Simple Network Time Protocol), versión simplificada de NTP
  - ~~Desventajas??~~



Implementación práctica



NTP server

RTC

- **Práctica 2**: Soporte de referencias de tiempo:
  - Integración de RTC y SNTP en servidor Web

❖ 1 semana

Ingeniería de Sistemas Electrónicos

Programación avanzada de microcontroladores (II)

- Integración de referencia temporal en el sistema
- Desarrollo de aplicaciones “confiables”
- Reducción del consumo
- Desarrollo de software. Control de versiones
- Prácticas de laboratorio

Desarrollo de aplicaciones "confiables"

- Alguna vez os ha fallado una aplicación que habéis desarrollado para un microcontrolador??

- Os suenan de algo estas frases???
  - Te juro que ayer me funcionaba!!!!
  - Uhmmm, creo que eso no lo había probado nunca…
  - Pues esto antes iba…
  - Se ha quedado "colgado", resetéalo
  - ….

Desarrollo de aplicaciones "confiables"

Los tres incorporan microcontroladores, pero…
¿dónde es más crítico un error?

Desarrollo de aplicaciones "confiables"

Desarrollo de aplicaciones “confiables”

Desarrollo de aplicaciones “confiables”

El seminario está enfocado tanto para aquellos que ya certifican o siguen esta metodología como para aquellos que aún no lo hacen pero que tienen interés en mejorar sus procesos o tienen previsión de empezar algún proyecto con requisitos de functional safety.

Tomando como punto de partida varias aplicaciones basadas en Arm Cortex-M, durante el seminario hablaremos de temas como:

- Entornos de desarrollo
- Compiladores certificados
- Paquetes específicos para certificac
- RTOS certificados
- Análisis estático
- Test de software
- Cobertura de código

http://www2.keil.com/fusa-rts
http://www2.keil.com/mdk5/safety?utm_campaign=arm_seminarios-safety-2020&utm_medium=email&utm_source=acumbamail
http://www2.keil.com/mdk5/verification-tools#lint
http://www2.keil.com/mdk5/debug/coverage

Desarrollo de aplicaciones "confiables"

Niveles de integridad de seguridad

Mapping of Risk to SIL	Severity of Consequence: Insignificant	Minor	Severe	Major	Catastrophic
Rare	-	-	SIL1	SIL2	SIL3
Unlikely	-	SIL1	SIL2	SIL3	SIL4
Likely	SIL1	SIL2	SIL3	SIL4	x
Very Likely	SIL2	SIL3	SIL4	x	x
Certain	SIL3	SIL4	x	x	x

Probability

Safety Integrity Levels

Industrial	IEC 61508	SIL 1	SIL 2		SIL 3
		Low			High
Automotive	ISO 26262	ASIL A	ASIL B	ASIL C	ASIL D
Railways	EN 50128	SIL 1	SIL 2	SIL 3	SIL 4
Medical	IEC 62304*	Class A	Class B		Class C

arm

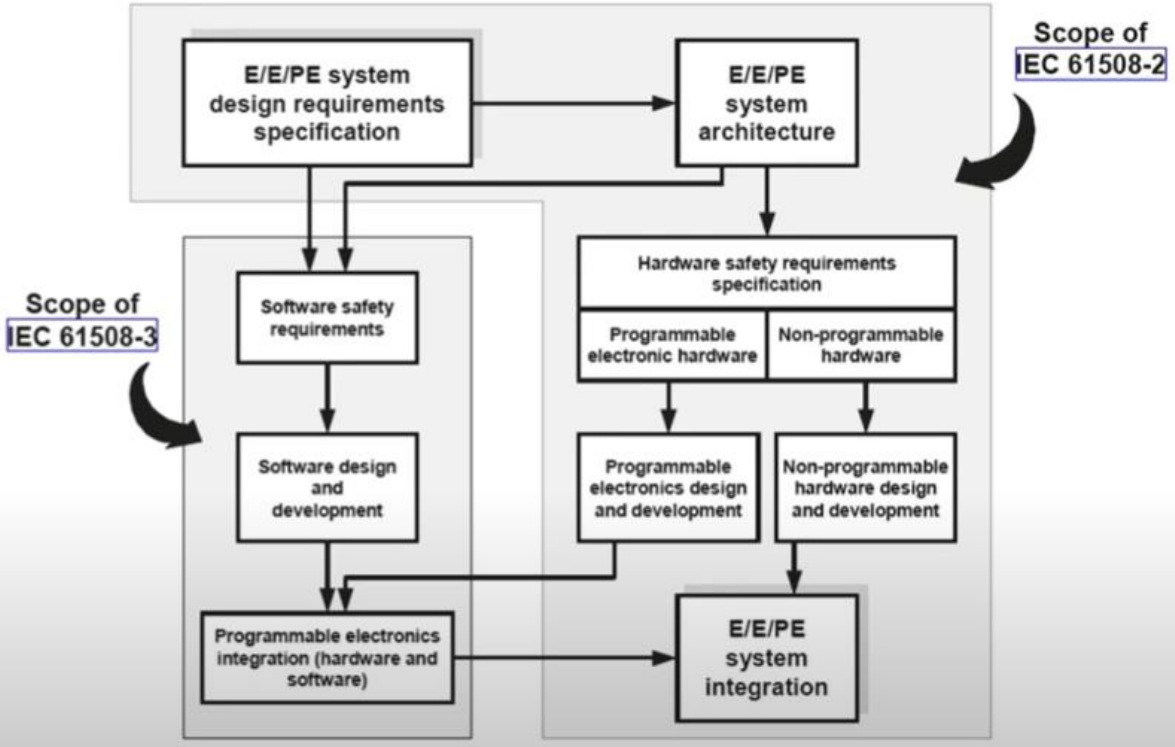
CAPEL
CAPTURA ELECTRÓNICA

Desarrollo de aplicaciones "confiables"

E/E/PE Safety System Lifecycle

Cada normativa define requisitos para:

- Procesos
- Documentación
- Hardware
- Software
- **Herramientas**
- ...



Desarrollo de aplicaciones "confiables"

Estrategias y herramientas "más a mano" que podemos utilizar para desarrollar aplicaciones más o menos confiables:

Estrategias:

- Buenas prácticas de desarrollo de código (uso de librerías, código estructurado, cumplimiento de guías de desarrollo, etc.)
- Uso de herramientas de depuración 
- Uso de herramientas de control de versiones
- Plan de test de los módulos independientes
- Test de código (code coverage?)
- …

Embedded C Coding Standard

https://barrgroup.com/embedded-systems/books/embedded-c-coding-standard

Desarrollo de aplicaciones "confiables"

Qué herramientas nos permiten determinar un fallo "no evidente" en el sistema:

Durante el desarrollo de la aplicación:

- Plan de test de cada módulo desarrollado
  - Análisis de especificaciones → funcionalidades a probar
  - Definir estrategia y tipos de pruebas. Criterios de aceptación
- Pruebas de integración
- Uso intensivo de herramientas de depuración
- Consulta de excepciones del sistema (¿?)
- …

Durante la "vida útil" de la aplicación:

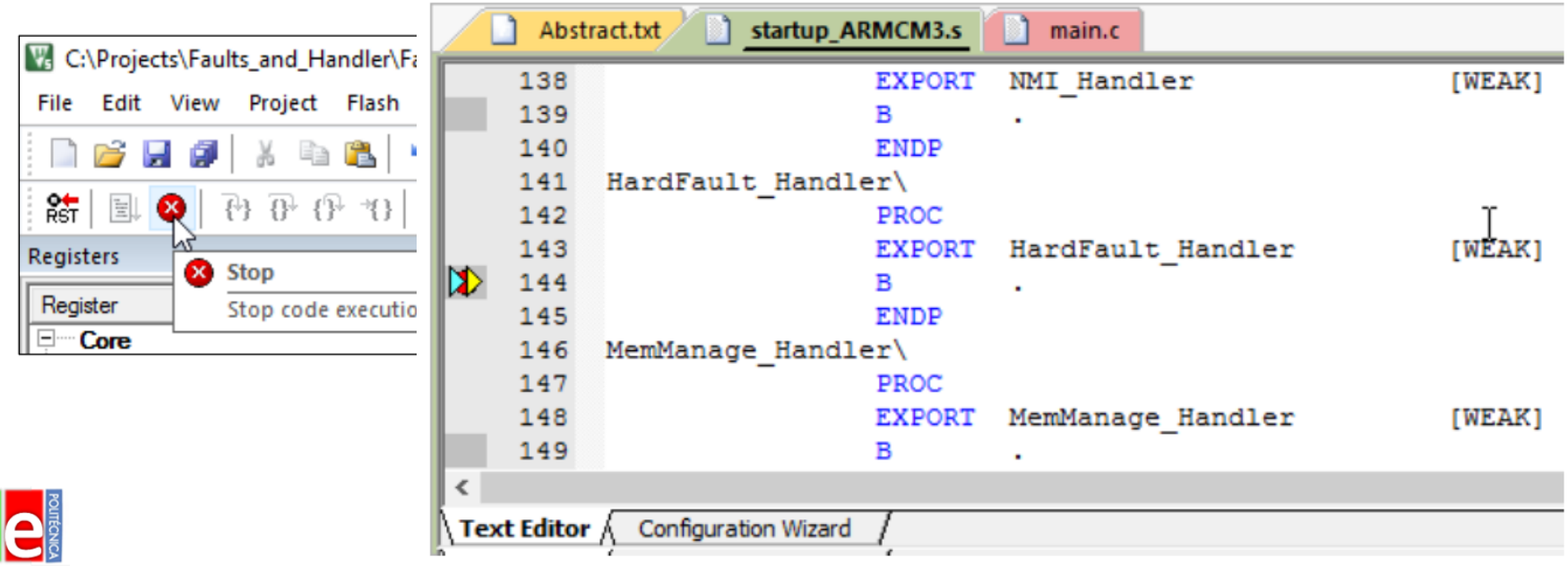
- Watchdog (Capítulos 21 y 22 del Reference Manual)
- Consulta de causa de reset del sistema (¿?)
- …

Fault Exceptions

Cuando una aplicación "se cuelga", es probable que se haya producido una *Fault Exception*.

Parar la ejecución del programa en el depurador y tratar de ver el tipo de excepción (y su causa) puede dar pistas del problema.

http://www.keil.com/appnotes/files/apnt209.pdf

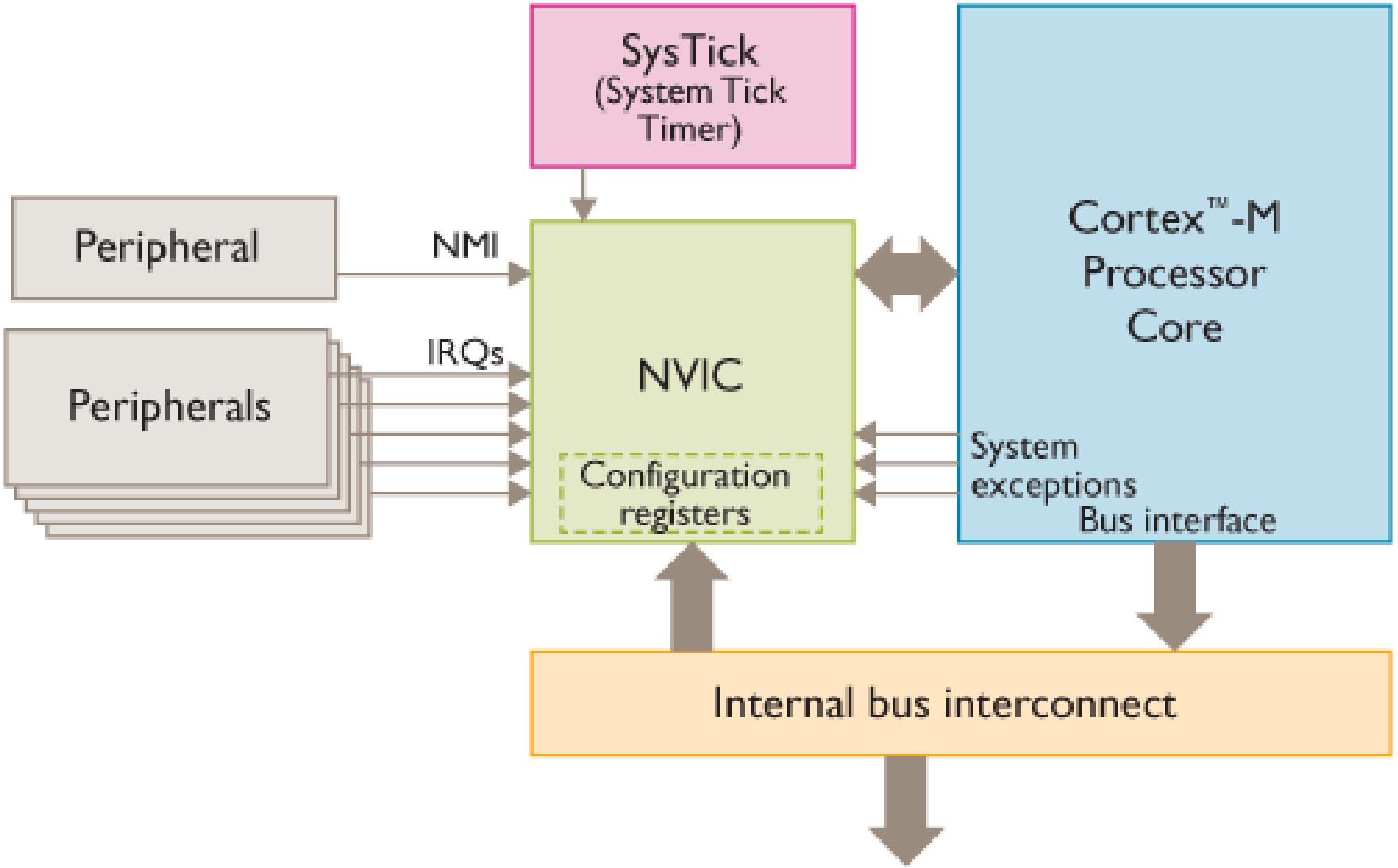


Excepciones

- What is an exception?
  - Any event (internal or external) that stops µP current process to switch to different task.
  - The exception priority is used to manage when it is attended by the µP, and to nest them.

- In Cortex-M
  - **System Exceptions** are numbered 1–15.
  - **External Interrupt** (to the µP, known as Interrupt Requests, IRQs) are numbered 16 and above.
  - Most of the exceptions have programmable priority, and a few have fixed priority.

NVIC (Nested Vector Interrupt Controller)



Excepciones

Exception Type	Name	Priority (Level Address)	Enable
1	Reset	−3	Always
2	NMI	−2	Always
3	Hard fault	−1	Always
4	MemManage	Programmable (0xE000ED18)	NVIC SHCSR (0xE000ED24) bit[16]
5	BusFault	Programmable (0xE000ED19)	NVIC SHCSR (0xE000ED24) bit[17]
6	Usage fault	Programmable (0xE000ED1A)	NVIC SHCSR (0xE000ED24) bit[18]
7–10	–	–	–
11	SVC	Programmable (0xE000ED1F)	Always
12	Debug monitor	Programmable (0xE000ED20)	NVIC DEMCR (0xE000EDFC) bit[16]
13	–	–	–
14	PendSV	Programmable (0xE000ED22)	Always
15	SysTick	Programmable (0xE000ED23)	SYSTICK CTRLSTAT (0xE000E010) bit[1]
16–255	IRQ	Programmable (0xE000E400)	NVIC SETEN (0xE000E100)

Fault exceptions

Fault Exceptions

- **Hard Fault Exception**
  - ✓ Usage faults, bus faults, and memory management faults if their handler cannot be executed
  - ✓ A bus fault during vector fetch
  - ✓ Hard Fault Status Register (HFSR) is used in the exception handler to determine the cause of the fault

- **Bus Fault Exception**
  - ✓ Bus faults are produced when an error response is received during a transfer on the AHB interfaces
  - ✓ Bus Fault Status Register (BFSR) is used in the exception handler to determine the cause of the fault

- **Memory Management Fault Exception**
  - ✓ Memory Management faults, as access to memory regions not defined in MPU setup, execute code from non executable memory regions, writing to read-only regions, …
  - ✓ Memory Management Fault Status Register (MMFSR) is used in the exception handler to determine the cause of the fault

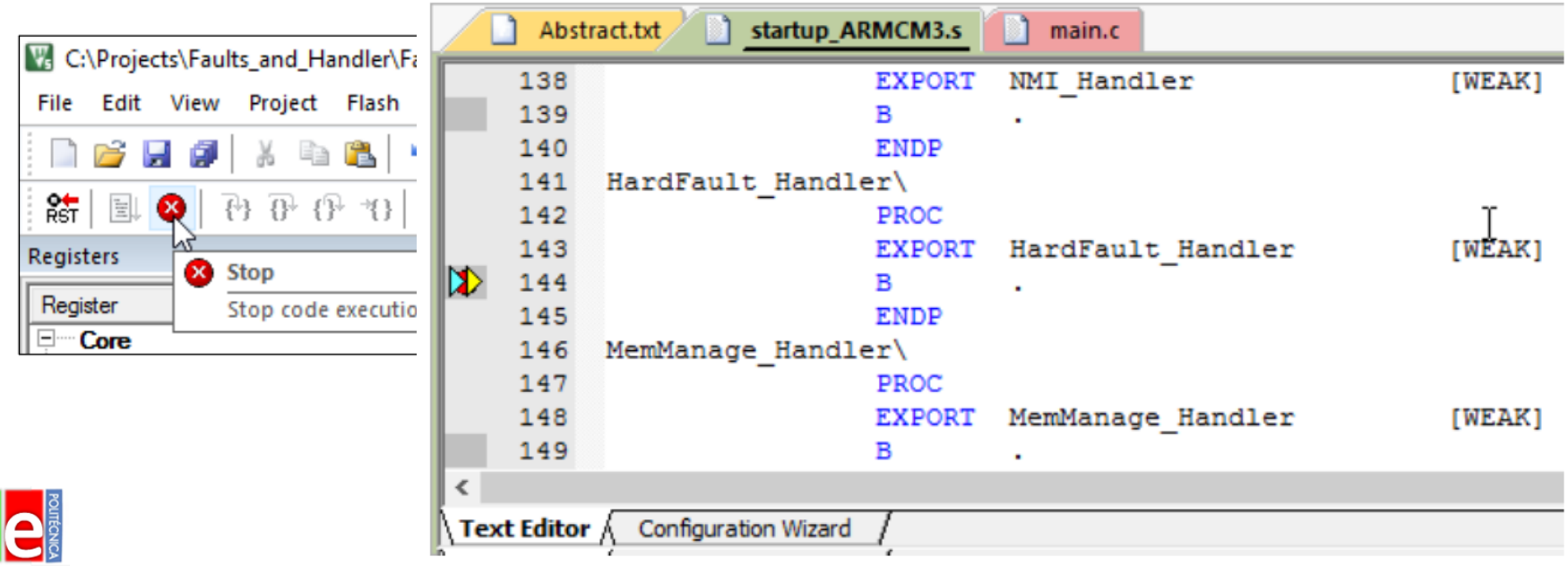
- **Usage Fault Exception**
  - ✓ Usage faults can be caused by undefined instructions, invalid interrupt return, divide by zero, etc.
  - ✓ Usage Fault Status Register (UFSR) is used in the exception handler to determine the cause of the fault

Fault Exceptions

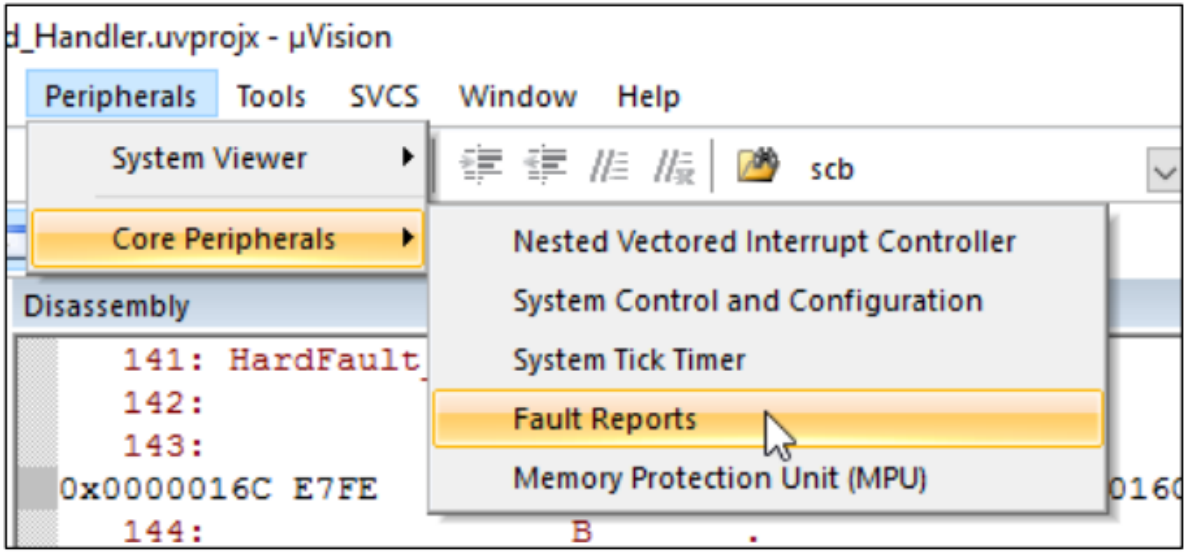
Cuando una aplicación “se cuelga”, es probable que se haya producido una *Fault Exception*.

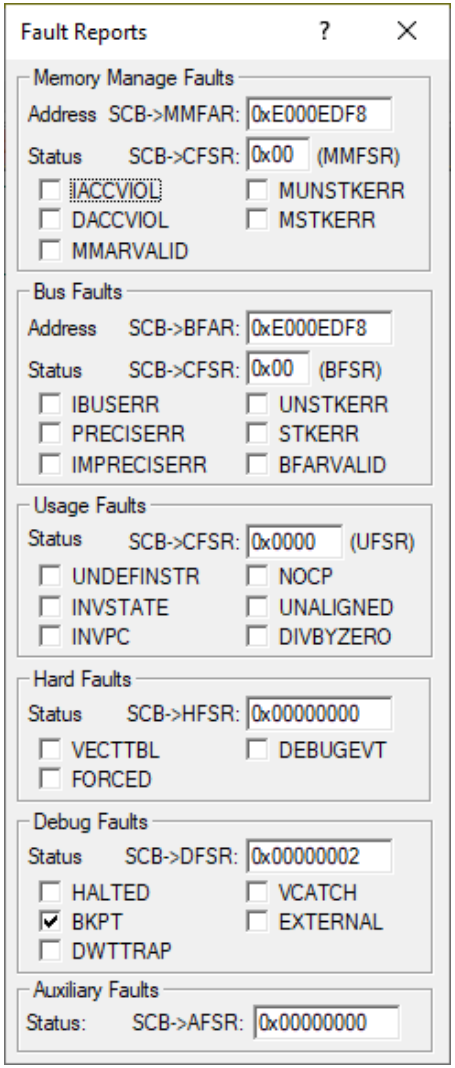
Parar la ejecución del programa en el depurador y tratar de ver el tipo de excepción (y su causa) puede dar pistas del problema.

http://www.keil.com/appnotes/files/apnt209.pdf



También es útil revisar la información del Fault Reports.





Fault Exceptions

stm32f4xx_it.c

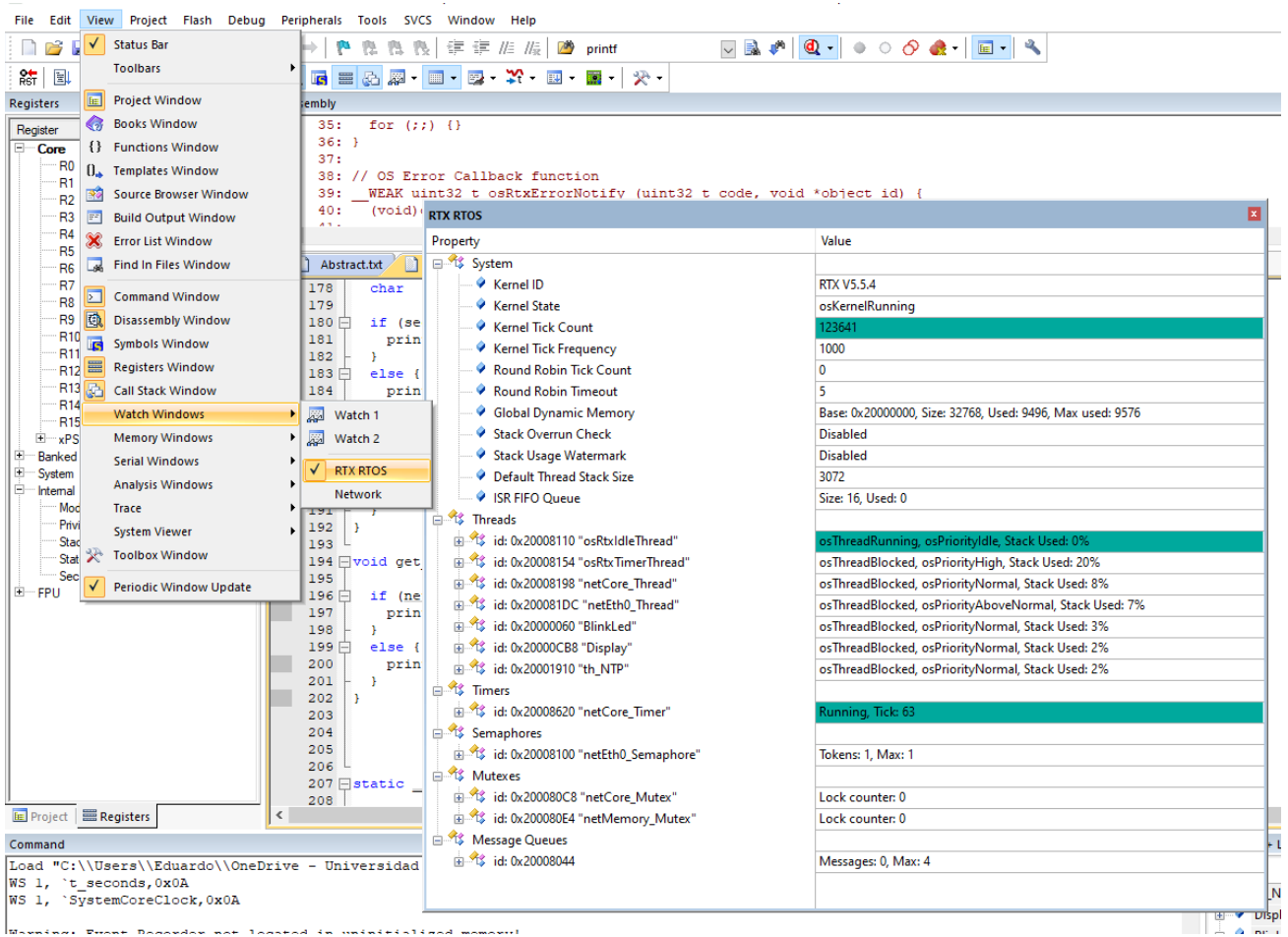
```
61  void HardFault_Handler(void)
62  {
63    /* Go to infinite loop when Hard Fault exception occurs */
64    while (1)
65    {
66    }
67  }
68
69  /**
70    * @brief  This function handles Memory Manage exception.
71    * @param  None
72    * @retval None
73    */
74  void MemManage_Handler(void)
75  {
76    /* Go to infinite loop when Memory Manage exception occurs */
77    while (1)
78    {
79    }
80  }
81
82  /**
83    * @brief  This function handles Bus Fault exception.
84    * @param  None
85    * @retval None
86    */
87  void BusFault_Handler(void)
88  {
89    /* Go to infinite loop when Bus Fault exception occurs */
90    while (1)
91    {
92    }
93  }
94
95  /**
96    * @brief  This function handles Usage Fault exception.
97    * @param  None
98    * @retval None
99    */
100 void UsageFault_Handler(void)
101 {
102   /* Go to infinite loop when Usage Fault exception occurs */
103   while (1)
104   {
105   }
```

...o incluir código en el fichero de manejadores de interrupciones *stm32f4xx_it.c.*

Problemas con el RTOS

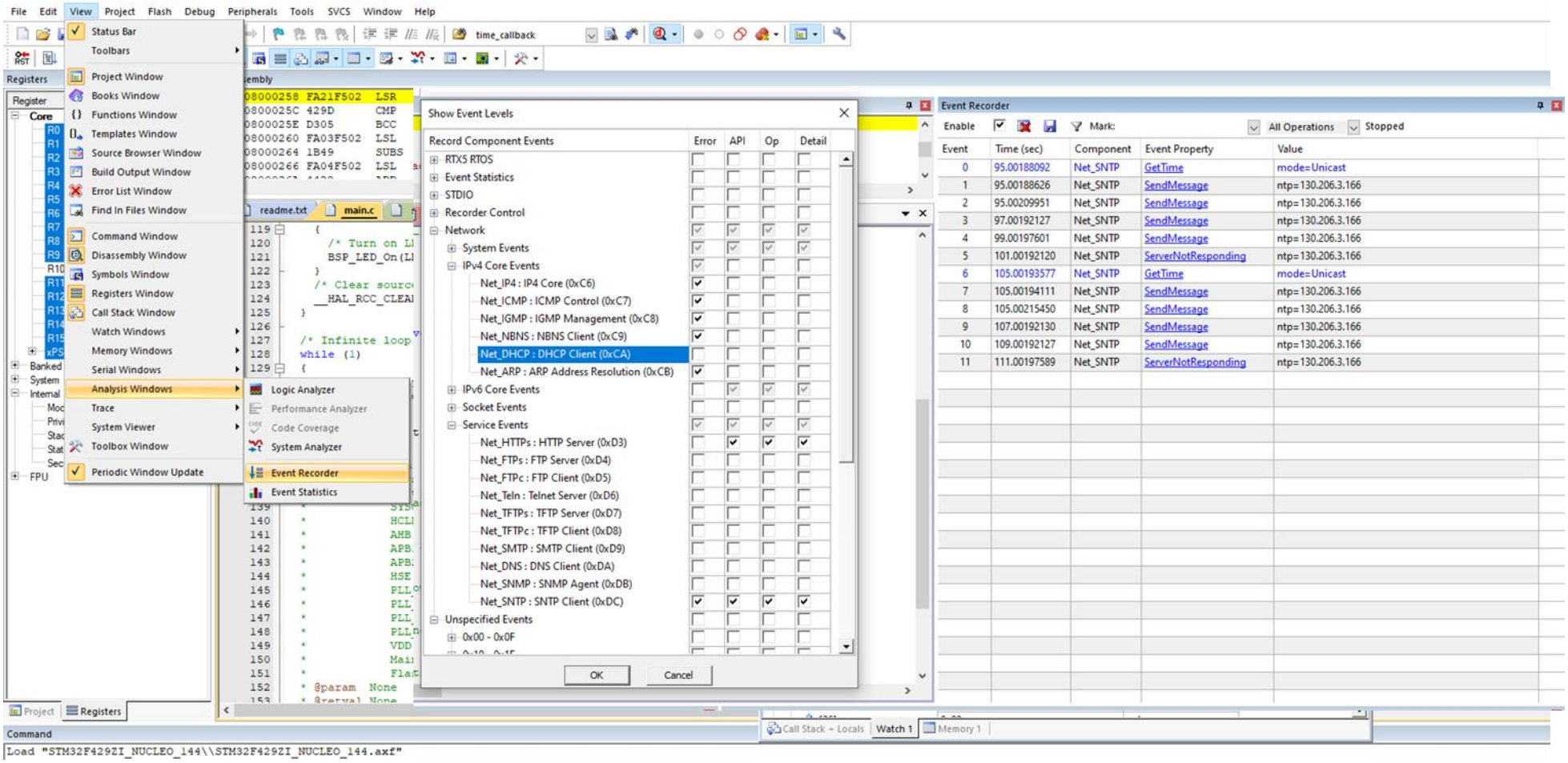
En ocasiones, el fallo se produce por una mala utilización de los recursos que proporciona el Sistema Operativo.

Utilizar las herramientas del entorno puede ayudar a detectar el problema

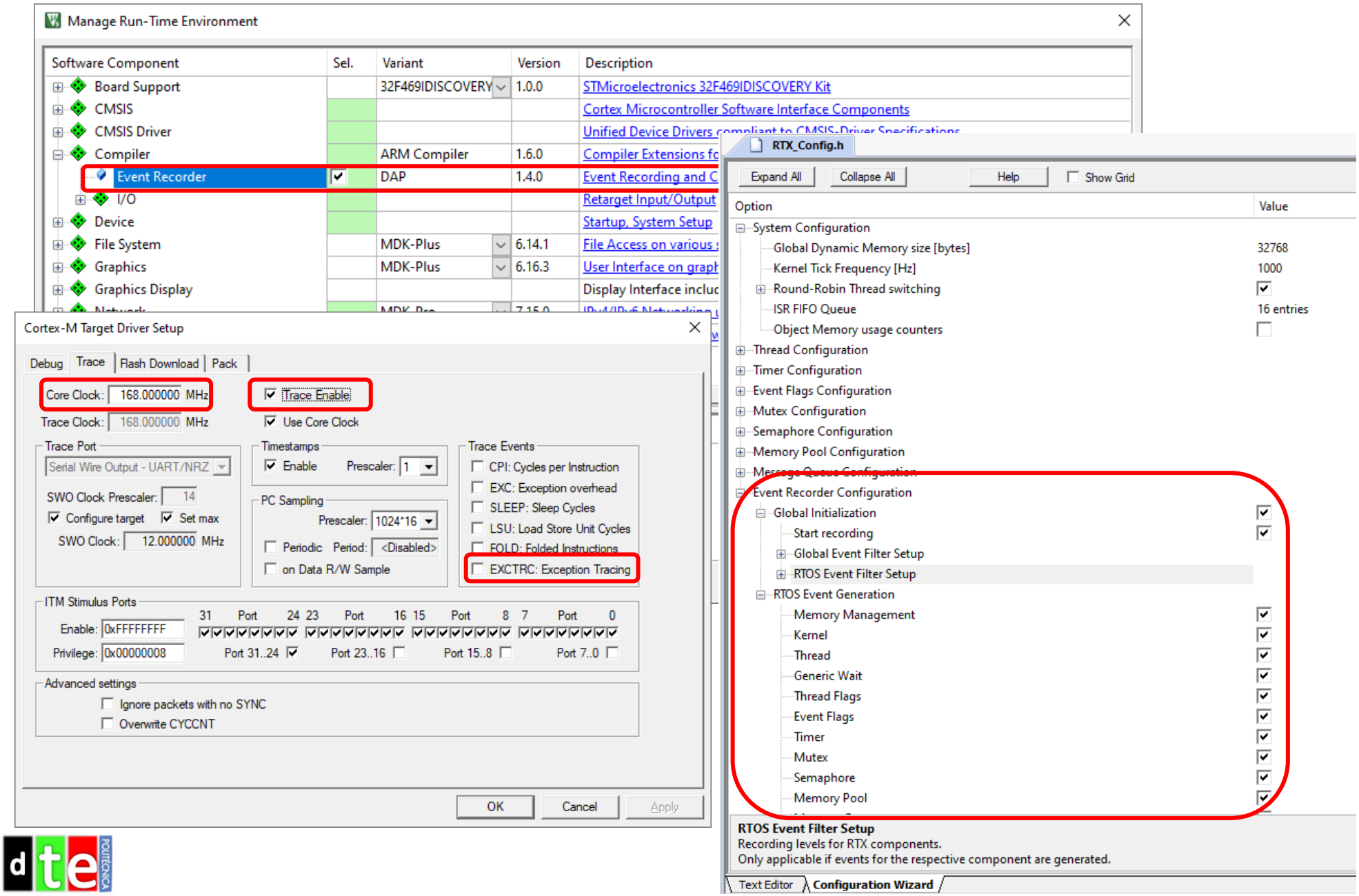


Event recorder

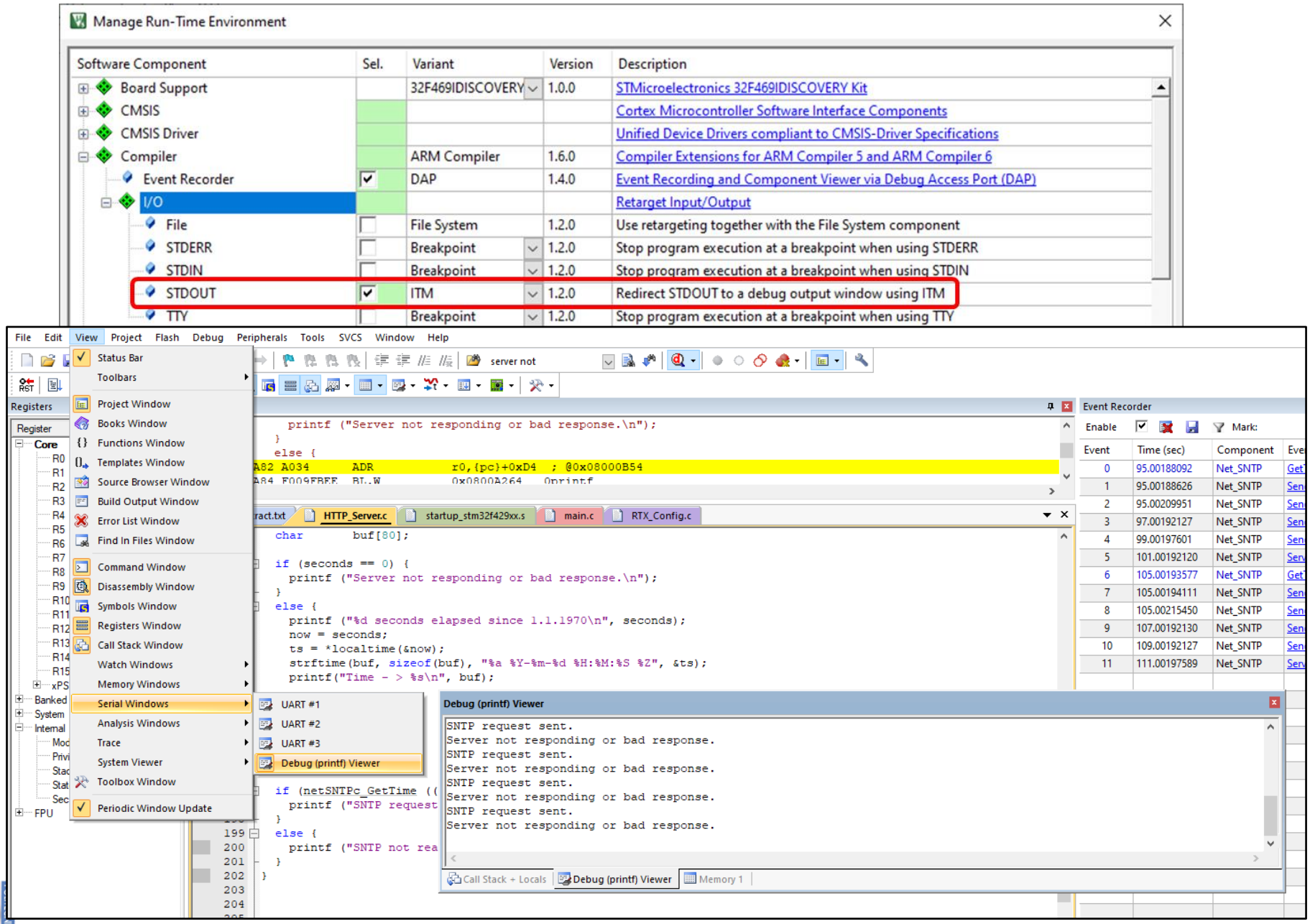
The software component **Event Recorder** provides an API (function calls) for event annotations in the application code or software component libraries. These API functions record event timing and data information while the program is executing.



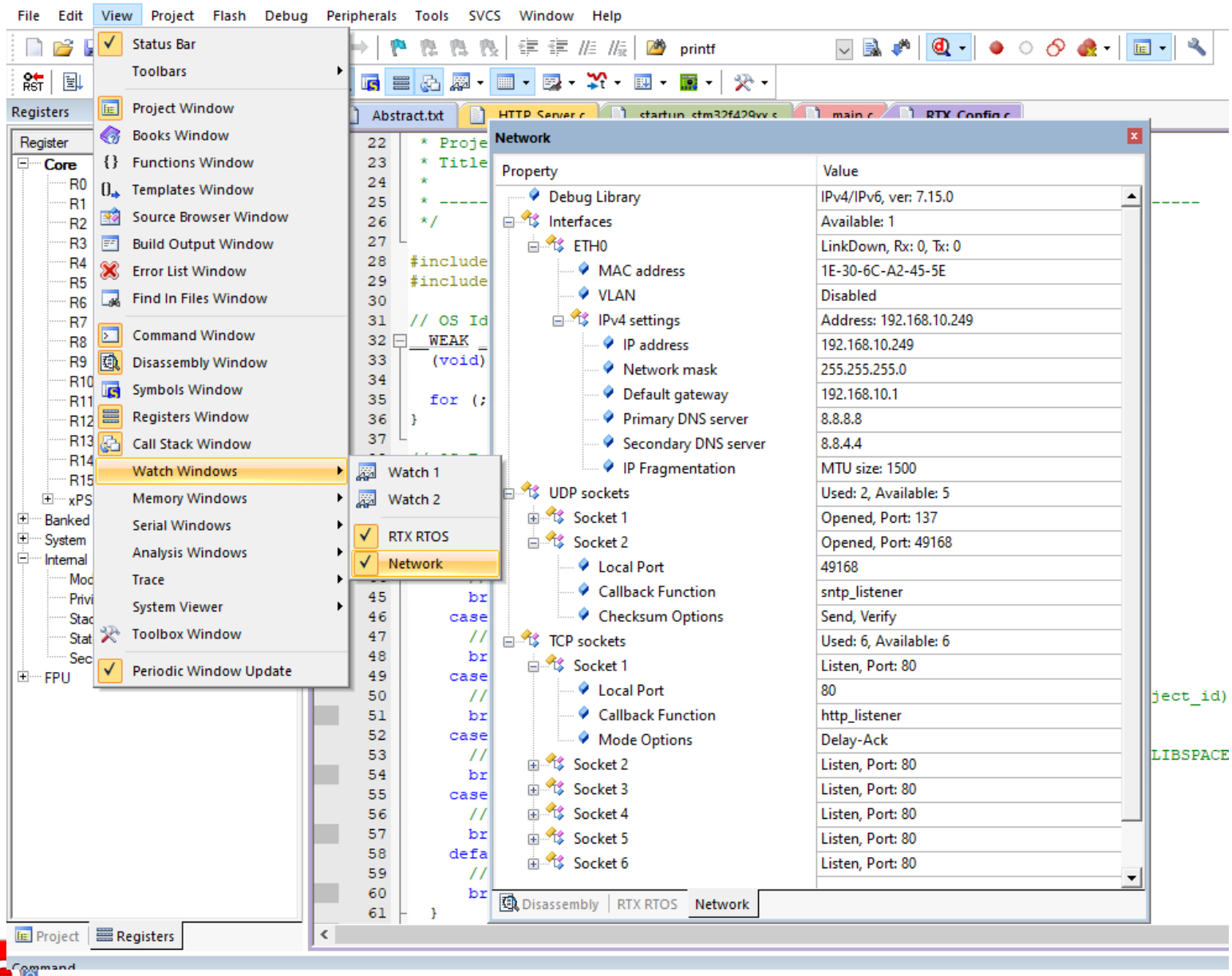
Event recorder setup



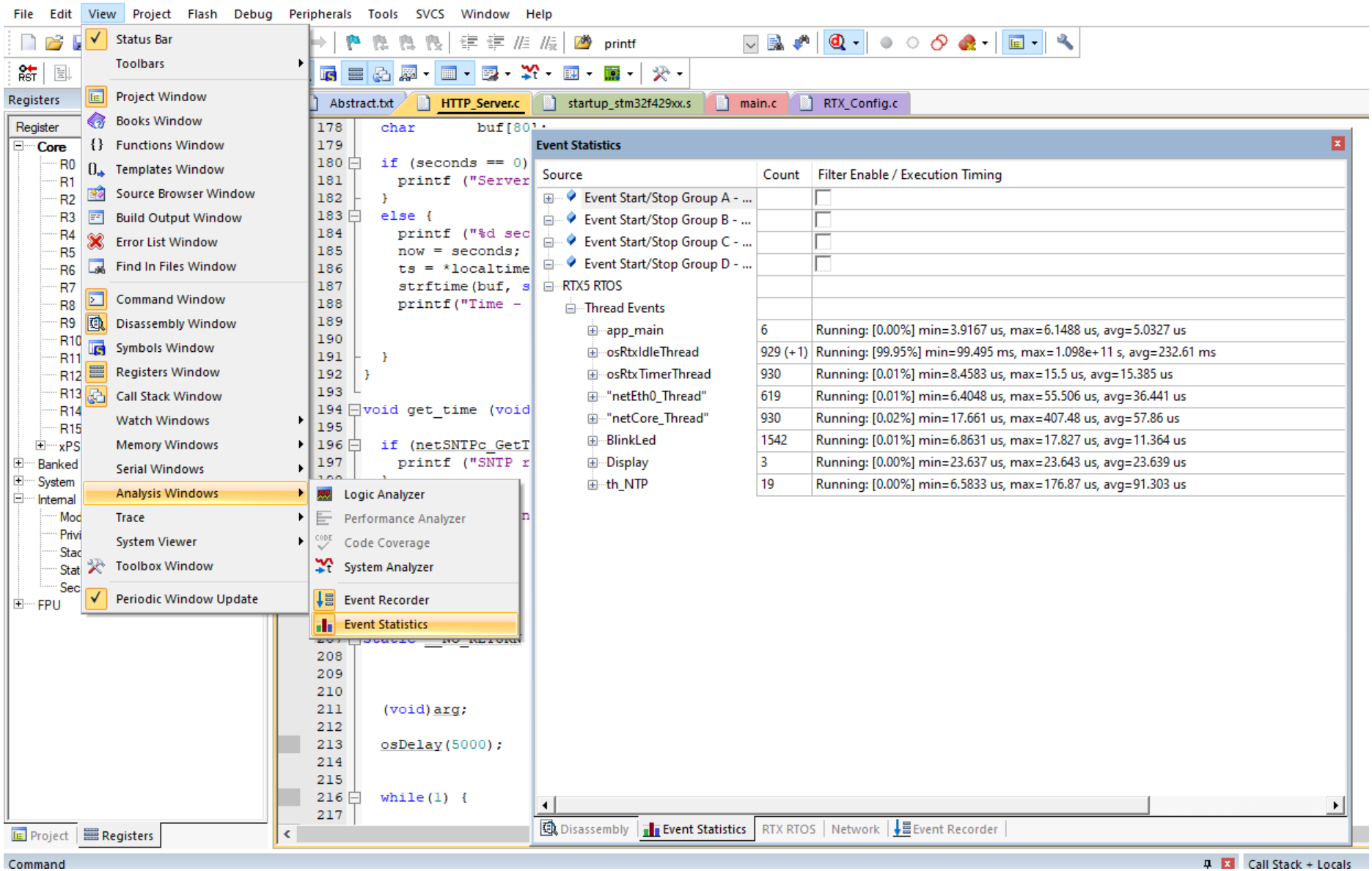
I/O Retargeting. Debug (printf) Viewer



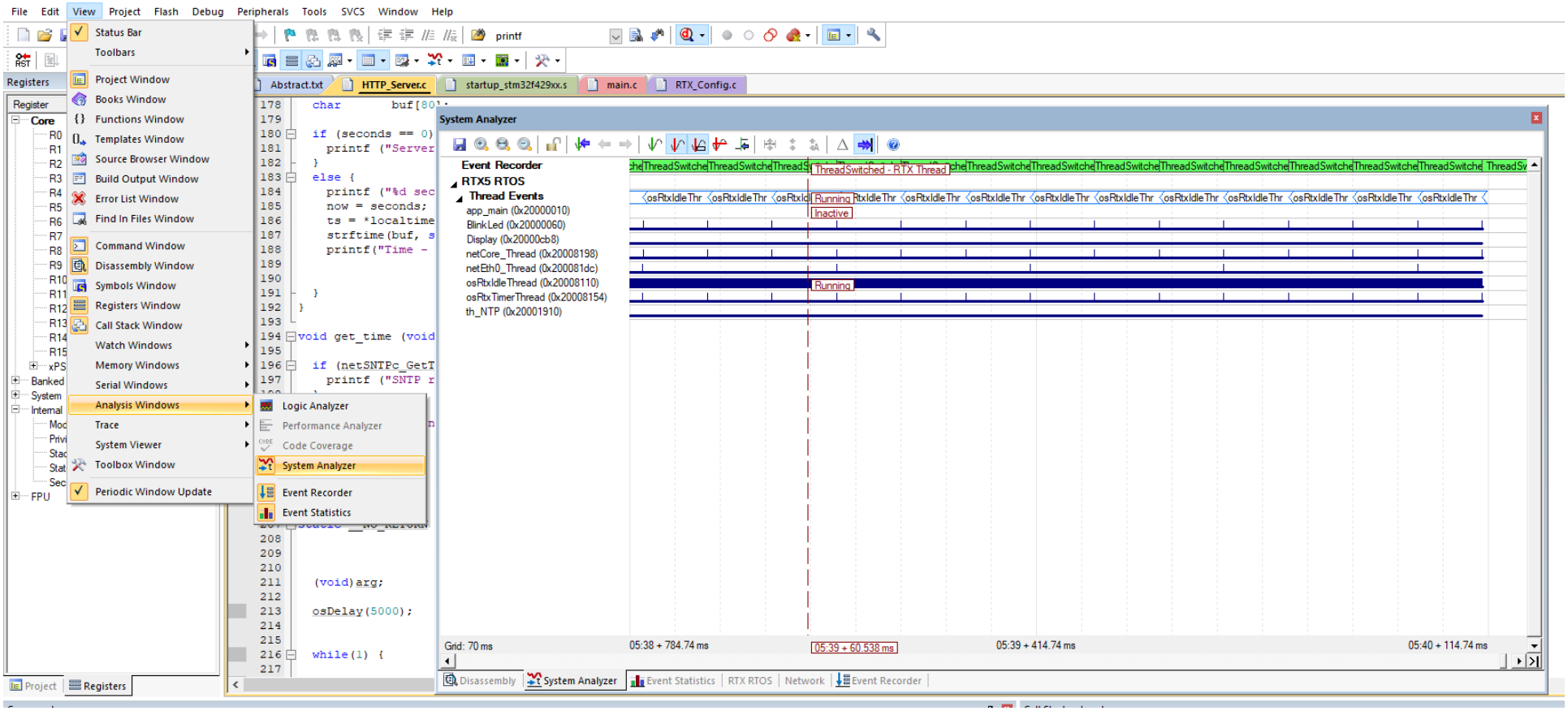
Network Watch Window



Event statistics



System Analyzer



Desarrollo de aplicaciones "confiables"

Qué herramientas nos permiten determinar un fallo "no evidente" en el sistema:

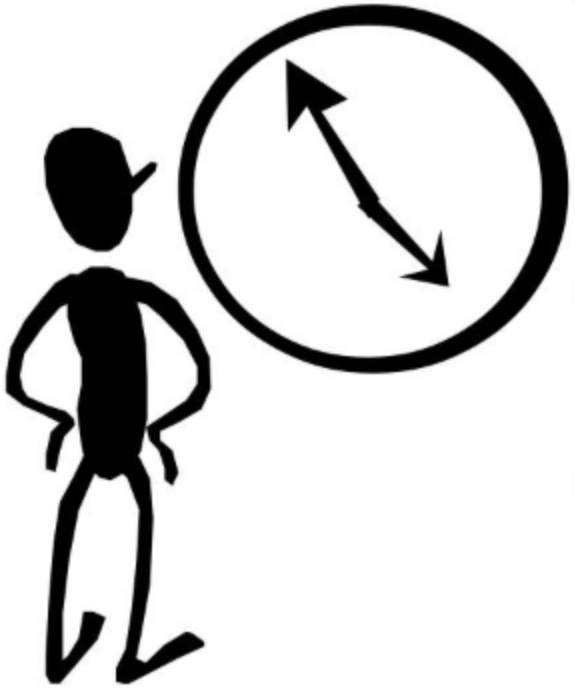
Durante el desarrollo de la aplicación:

- Plan de test de cada módulo desarrollado
  - Análisis de especificaciones → funcionalidades a probar
  - Definir estrategia y tipos de pruebas. Criterios de aceptación
- Pruebas de integración
- Uso intensivo de herramientas de depuración
- Consulta de excepciones del sistema (¿?)
- …

Durante la "vida útil" de la aplicación:

- Watchdog (Capítulos 21 y 22 del Reference Manual)
- Consulta de causa de reset del sistema (¿?)
- …

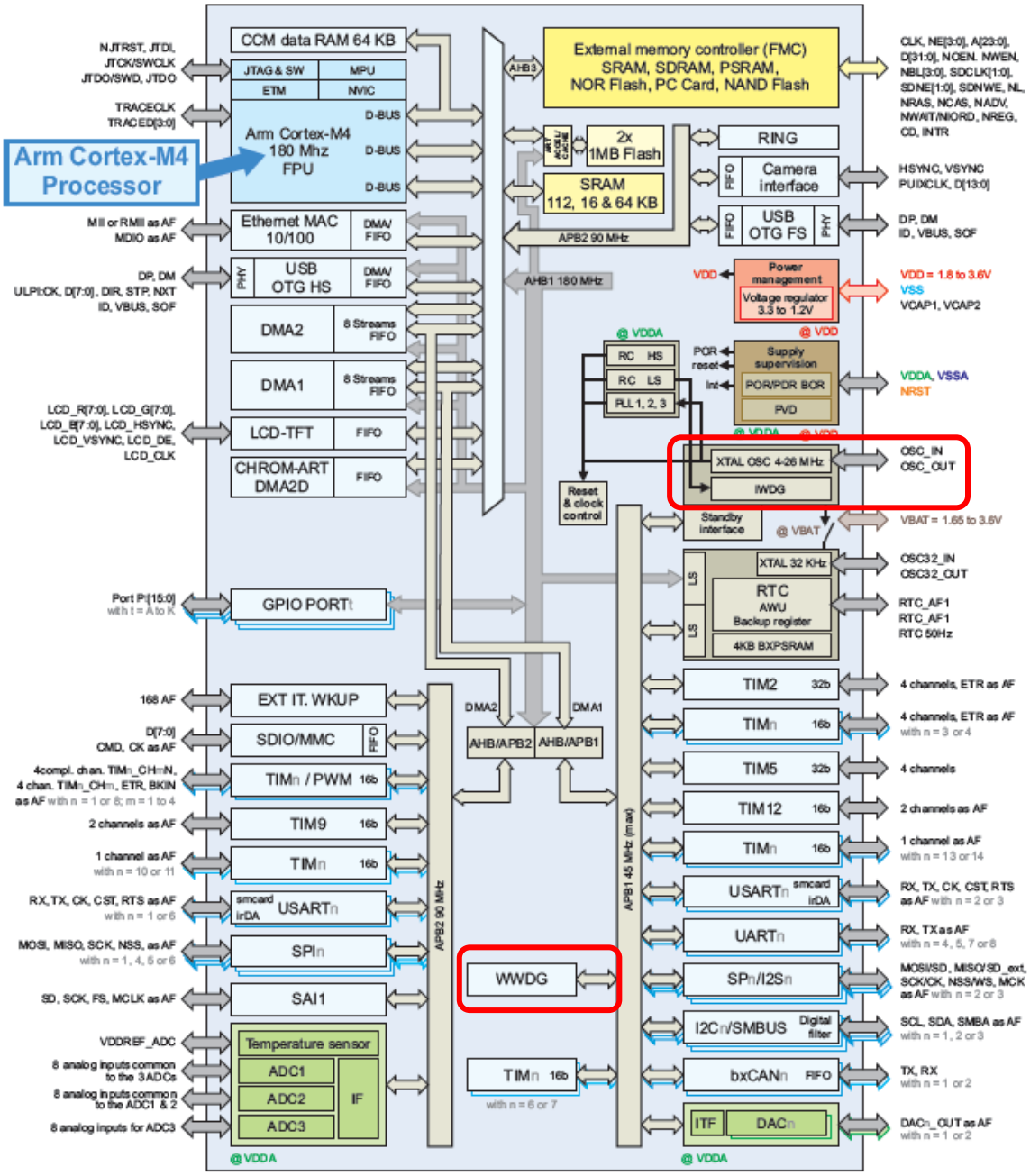
What is the watchdog timer?



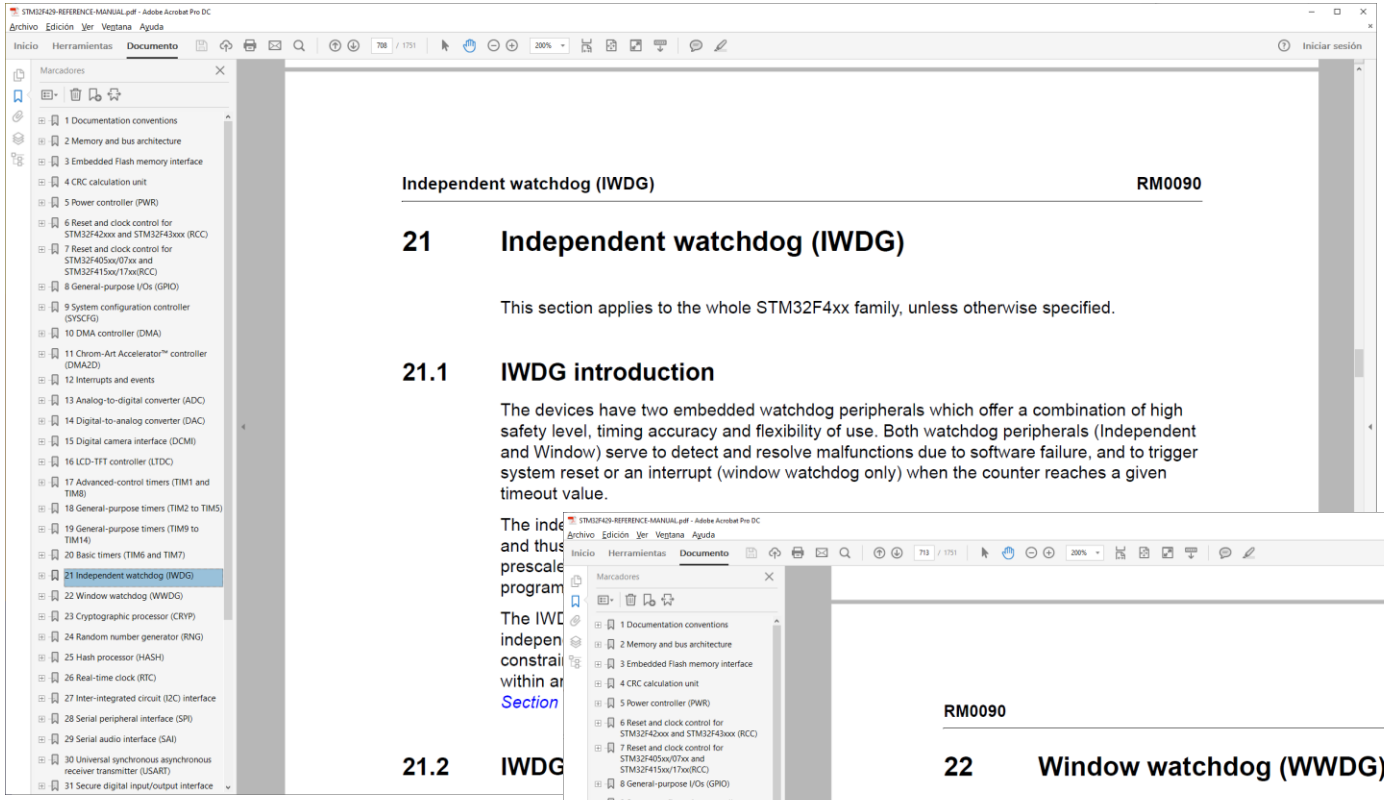
- A special timer that, once configured and enabled, counts down to 0
- When expired, causes the CPU to reset
- The countdown can be restarted at any time before expiration
- If periodically restarted, the expiration will never occur, and the reset will never happen.

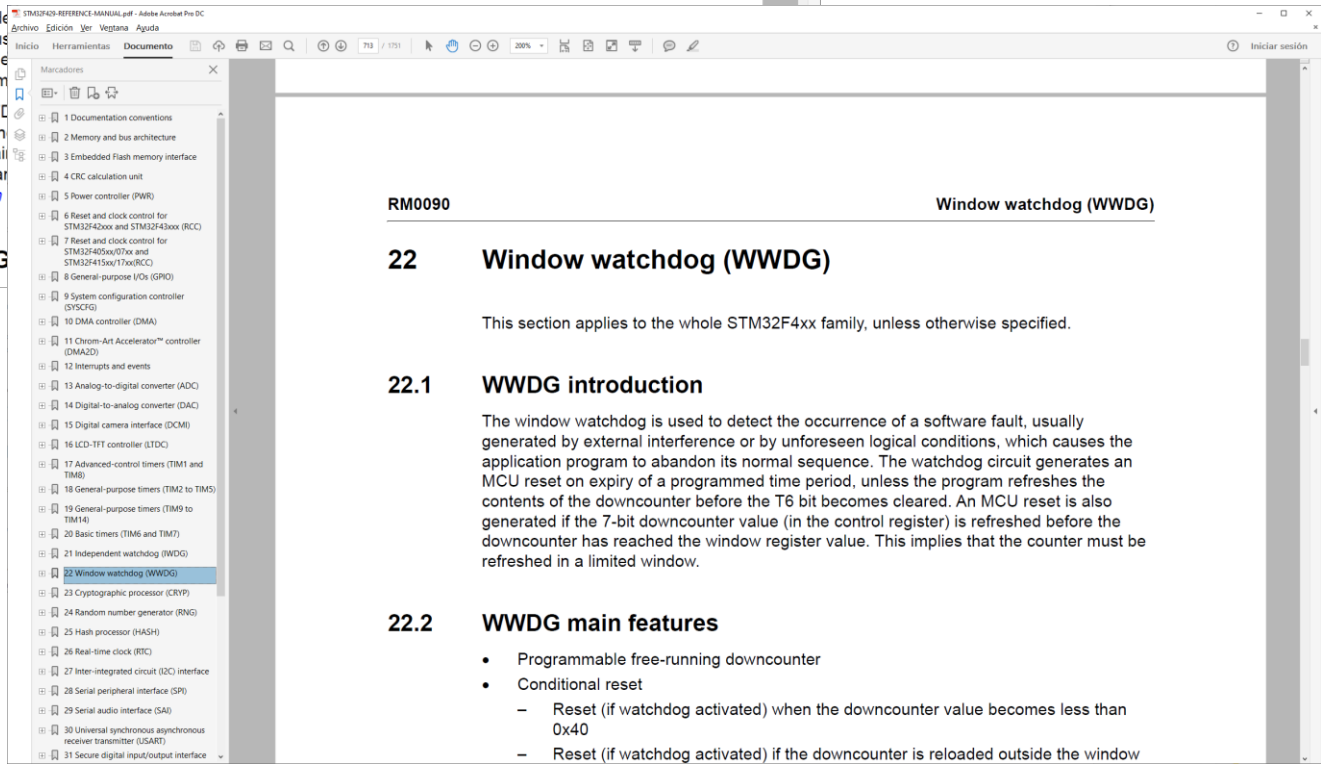
Watchdog

Timer/s que “vigilan” el sistema. Ante un comportamiento “anormal”, puede provocar bien una **interrupción**, bien un **reset del sistema**.



Watchdogs en STM32F4





- **Práctica 3**: Modos de bajo consumo y Watchdog
  - Apartado 4 (opcional) Watchdog e identificación de causa de reset del sistema

❖ 1 semana

Ingeniería de Sistemas Electrónicos

Programación avanzada de microcontroladores (II)

- Integración de referencia temporal en el sistema
- Desarrollo de aplicaciones "confiables"
- Reducción del consumo
- Desarrollo de software. Control de versiones
- Prácticas de laboratorio

Modos de bajo consumo en STM32F4

5 Power controller (PWR)

5.3 Low-power modes

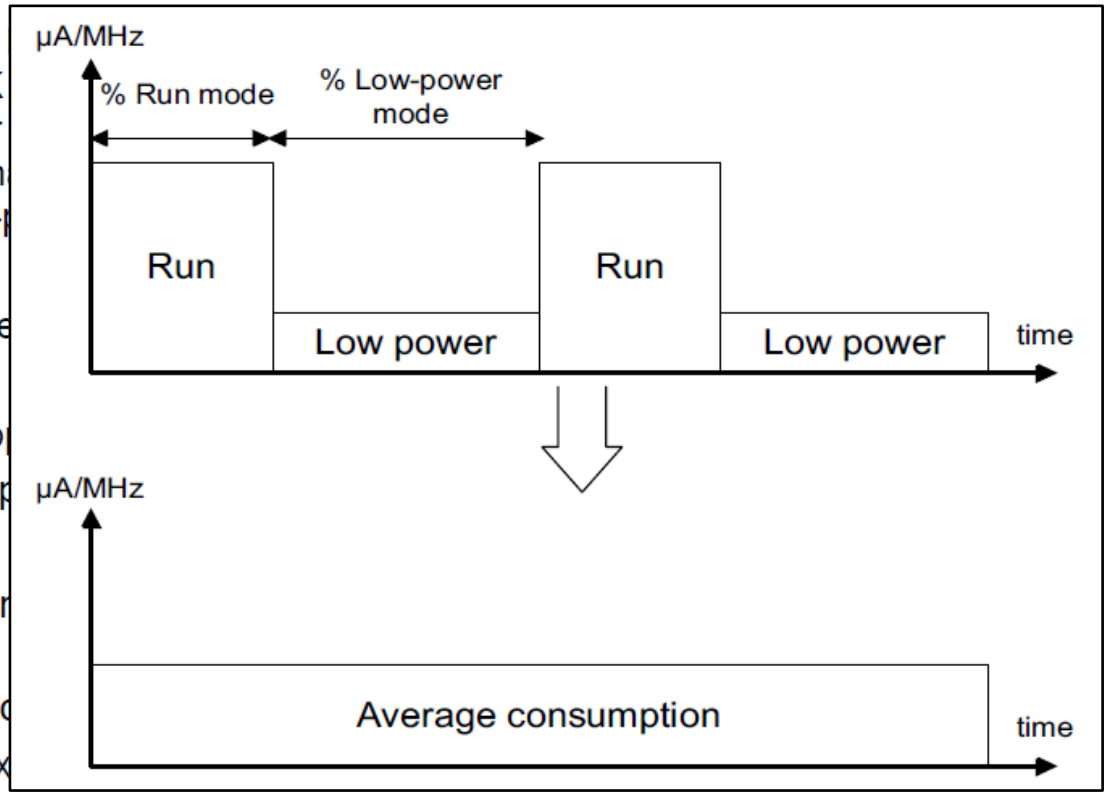
By default, the microcontroller is in
mode the CPU is clocked by HCLK
modes are available to save power
example when waiting for an extern
the best compromise between low-p
wakeup sources.

The devices feature three low-powe

- Sleep mode (Cortex®-M4 with
- Stop mode (all clocks are stop
- Standby mode (1.2 V domain p

In addition, the power consumption
means:

- Slowing down the system cloc
- Gating the clocks to the APBx



Modos de bajo consumo en STM32F4

life.augmented

AN4365
Application note

Using STM32F4 MCU power modes with best dynamic efficiency

https://www.st.com/resource/en/application_note/an4365-using-stm32f4-mcu-power-modes-with-best-dynamic-efficiency-stmicroelectronics.pdf

STM32F4 devices feature four main low-power modes:

- **Sleep mode:**
  - Only the CPU clock is stopped.

  The Cortex-M4 clock is stopped and the peripherals are kept running. The current consumption increases with the clock frequency. As in Run mode, the user should be aware of system configuration rules that concern the system clock and voltage regulator scales.
- **Stop mode:**
  - Lowest power consumption while all the SRAM and registers are kept.
  - PLL, HSI, HSE are disabled.
  - All clocks in 1.2 V domain are switched-off.
  - Voltage regulator is working in Normal mode or Low-power mode.
  - Flash memory is working in Stop mode or Deep-power down mode.

  The Cortex-M4 core is stopped and the clocks are switched off (the PLL, the HSI and the HSE are disabled). SRAM and register content are kept. All I/O pins keep the same state as the Run mode. The voltage regulator is working in Normal or Low-power mode and the Flash memory can be configured in Stop mode or Deep-power down mode to save more static power.

Modos de bajo consumo en STM32F4

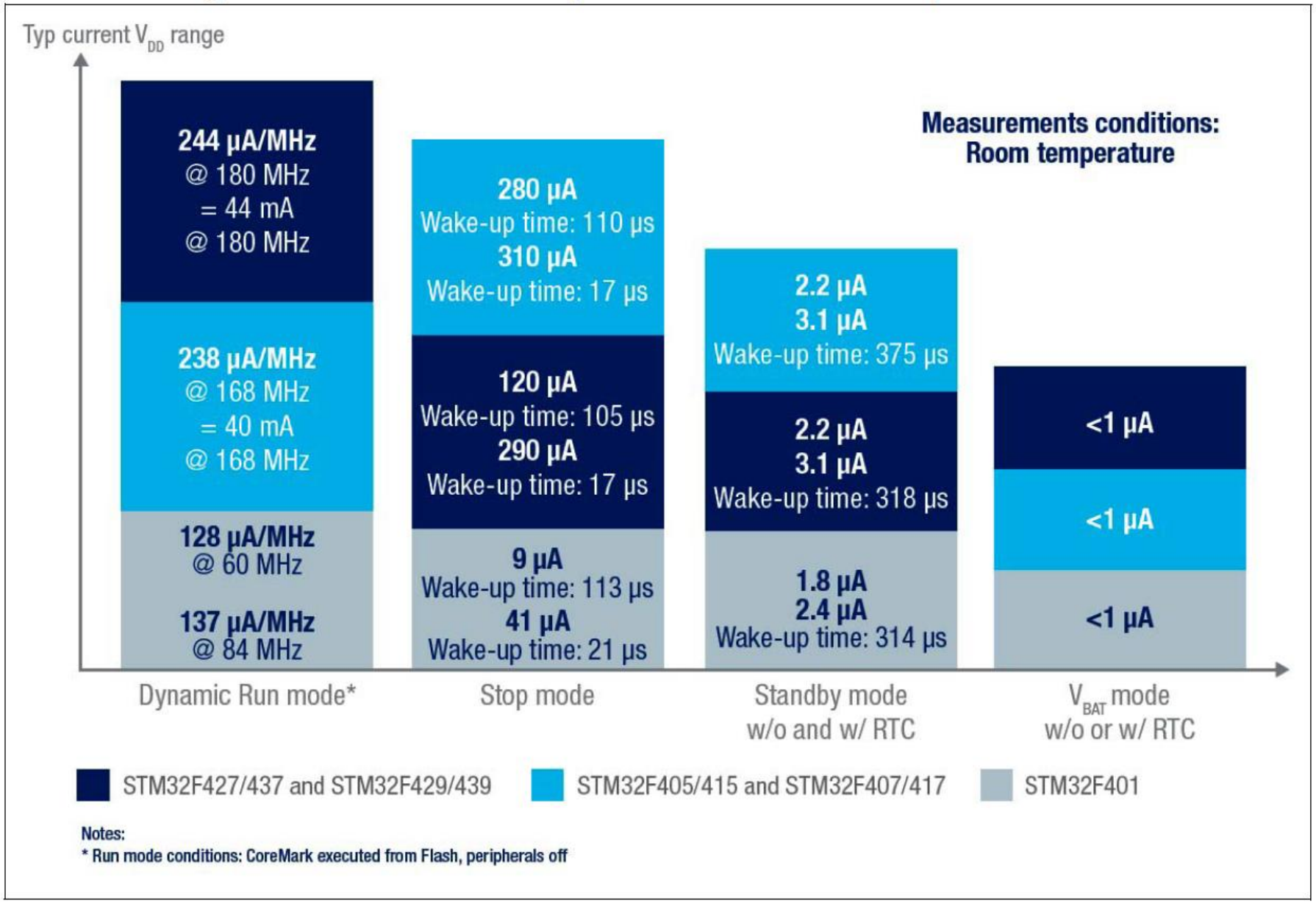
- **Standby mode:**
  - The lowest power consumption.
  - The 1.2 domain is powered off (regulator is disabled).
  - SRAM and register contents are lost except in the backup domain.

  The Cortex-M4 core is stopped and the clocks are switched off. The voltage regulator is disabled and the 1.2 V domain is powered-off. SRAM and register contents are lost except for registers in the Backup domain (RTC registers, RTC backup register and backup SRAM), and Standby circuitry. This mode has the lowest current consumption, which depends on the configuration of both backup SRAM (not available for STM32F401x) and RTC.
- **V_{BAT} mode**:
  - The main digital supply (V_{DD}) is turned off.
  - The circuit is supplied through V_{BAT} pin which should be connected to an external supply voltage (a battery or any other source).

  This mode is used only when the main digital supply (V_{DD}) is turned off and the V_{BAT} pin is connected to an external supply voltage (a battery or any other source). The V_{BAT} pin powers the Backup domain (RTC registers, RTC backup register and backup SRAM).

Modos de bajo consumo en STM32F4

Figure 2. Current consumption of STM32F4 MCU power modes



Notes:
* Run mode conditions: CoreMark executed from Flash, peripherals off

Modos de bajo consumo. Entrada

Entering low-power mode

Low-power modes are entered by the MCU by executing the WFI (Wait For Interrupt), or WFE (Wait for Event) instructions, or when the SLEEPONEXIT bit in the Cortex®-M4 with FPU System Control register is set on Return from ISR.

Entering Low-power mode through WFI or WFE will be executed only if no interrupt is pending or no event is pending.

WFI (Wait For Interrupt) instruction → `__WFI();`
WFE (Wait For Exception) instruction → `__WFE();`

- El uso de las instrucciones anteriores hace que el "programa principal" se detenga a la espera de una interrupción (o una excepción)
- Útil para las denominadas "interrupt driven applications"

- Nota de aplicación "*AN3430. How to achieve the lowest current consumption with STM32F2xx*"

https://www.st.com/resource/en/application_note/an3430-how-to-achieve-the-lowest-current-consumption-with-stm32f2xx-stmicroelectronics.pdf

Modos de bajo consumo. Salida

Exiting low-power mode

The MCU exits from Sleep and Stop modes low-power mode depending on the way the low-power mode was entered:

- If the WFI instruction or Return from ISR was used to enter the low-power mode, any peripheral interrupt acknowledged by the NVIC can wake up the device.
- If the WFE instruction is used to enter the low-power mode, the MCU exits the low-power mode as soon as an event occurs. The wakeup event can be generated either by:
  - NVIC IRQ interrupt:

    When SEVONPEND = 0 in the Cortex®-M4 with FPU System Control register: by enabling an interrupt in the peripheral control register and in the NVIC. When the MCU resumes from WFE, the peripheral interrupt pending bit and the NVIC peripheral IRQ channel pending bit (in the NVIC interrupt clear pending register) have to be cleared. Only NVIC interrupts with sufficient priority will wakeup and interrupt the MCU.

    When SEVONPEND = 1 in the Cortex®-M4 with FPU System Control register: by enabling an interrupt in the peripheral control register and optionally in the NVIC. When the MCU resumes from WFE, the peripheral interrupt pending bit and when enabled the NVIC peripheral IRQ channel pending bit (in the NVIC interrupt clear pending register) have to be cleared. All NVIC interrupts will wakeup the MCU, even the disabled ones.Only enabled NVIC interrupts with sufficient priority will wakeup and interrupt the MCU.
  - Event

    This is done by configuring a EXTI line in event mode. When the CPU resumes from WFE, it is not necessary to clear the EXTI peripheral interrupt pending bit or the NVIC IRQ channel pending bit as the pending bits corresponding to the event line is not set. It may be necessary to clear the interrupt flag in the peripheral.

The MCU exits from Standby low-power mode through an external reset (NRST pin), an IWDG reset, a rising edge on one of the enabled WKUPx pins or a RTC event occurs (see *Figure 237: RTC block diagram*).

After waking up from Standby mode, program execution restarts in the same way as after a Reset (boot pin sampling, option bytes loading, reset vector is fetched, etc.).

Modos de bajo consumo. Resumen

Table 23. Low-power mode summary

Mode name	Entry	Wakeup	Effect on 1.2 V domain clocks	Effect on V_{DD} domain clocks	Voltage regulator
Sleep (Sleep now or Sleep-on-exit)	WFI or Return from ISR	Any interrupt	CPU CLK OFF no effect on other clocks or analog clock sources	None	ON
	WFE	Wakeup event			
Stop	PDDS and LPDS bits + SLEEPDEEP bit + WFI, Return from ISR or WFE	Any EXTI line (configured in the EXTI registers, internal and external lines)	All 1.2 V domain clocks OFF	HSI and HSE oscillators OFF	ON or in low- power mode (depends on *PWR power control register (PWR_CR) for STM32F405xx/07xx and STM32F415xx/17xx* and *PWR power control register (PWR_CR) for STM32F405xx/07xx and STM32F415xx/17xxPWR power control register (PWR_CR) for STM32F42xxx and STM32F43xxx*
Standby	PDDS bit + SLEEPDEEP bit + WFI, Return from ISR or WFE	WKUP pin rising edge, RTC alarm (Alarm A or Alarm B), RTC Wakeup event, RTC tamper events, RTC time stamp event, external reset in NRST pin, IWDG reset			OFF

Otras recomendaciones para reducir el consumo

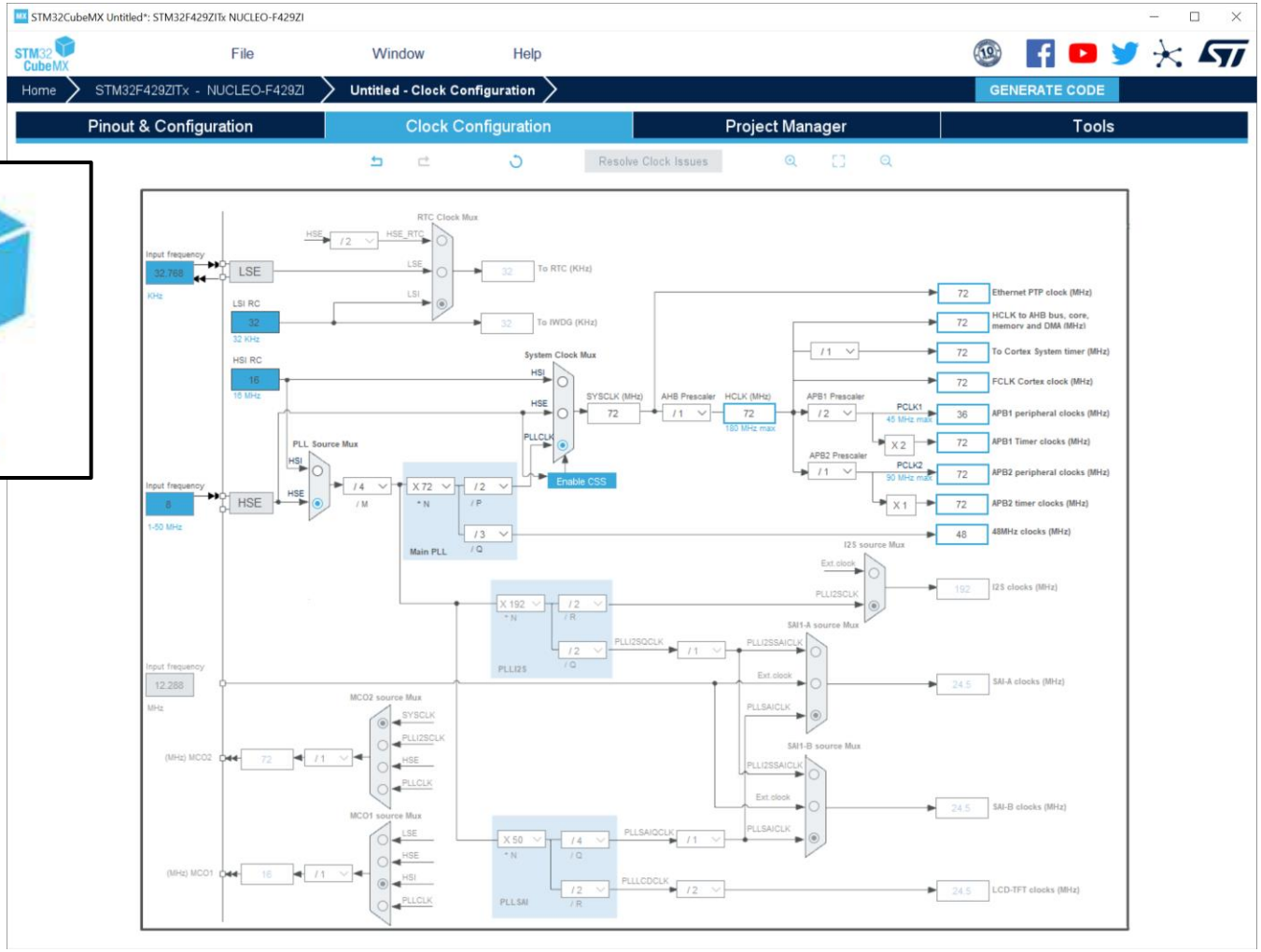
- Reducir al máximo la frecuencia de reloj del sistema
  - Algunos periféricos (Ethernet, USB, I2S and SDIO) necesitan frecuencias específicas de reloj que limitan el valor mínimo de la frecuencia de reloj del sistema
- Reducir al máximo la frecuencia de reloj de los periféricos
- Desactivar todos los periféricos no utilizados
- Configurar todos los pines del GPIO no utilizados (configurados por defecto como entradas digitales) como entradas analógicas (AIN)

- IMPORTANTE:
  - Cuando se hace entrar al sistema en un modo de bajo consumo no pueden utilizarse las herramientas de depuración (el core del Cortex-M4 no tiene reloj durante el modo de bajo consumo, por lo que las funciones de depuración están desactivadas)

Permitido

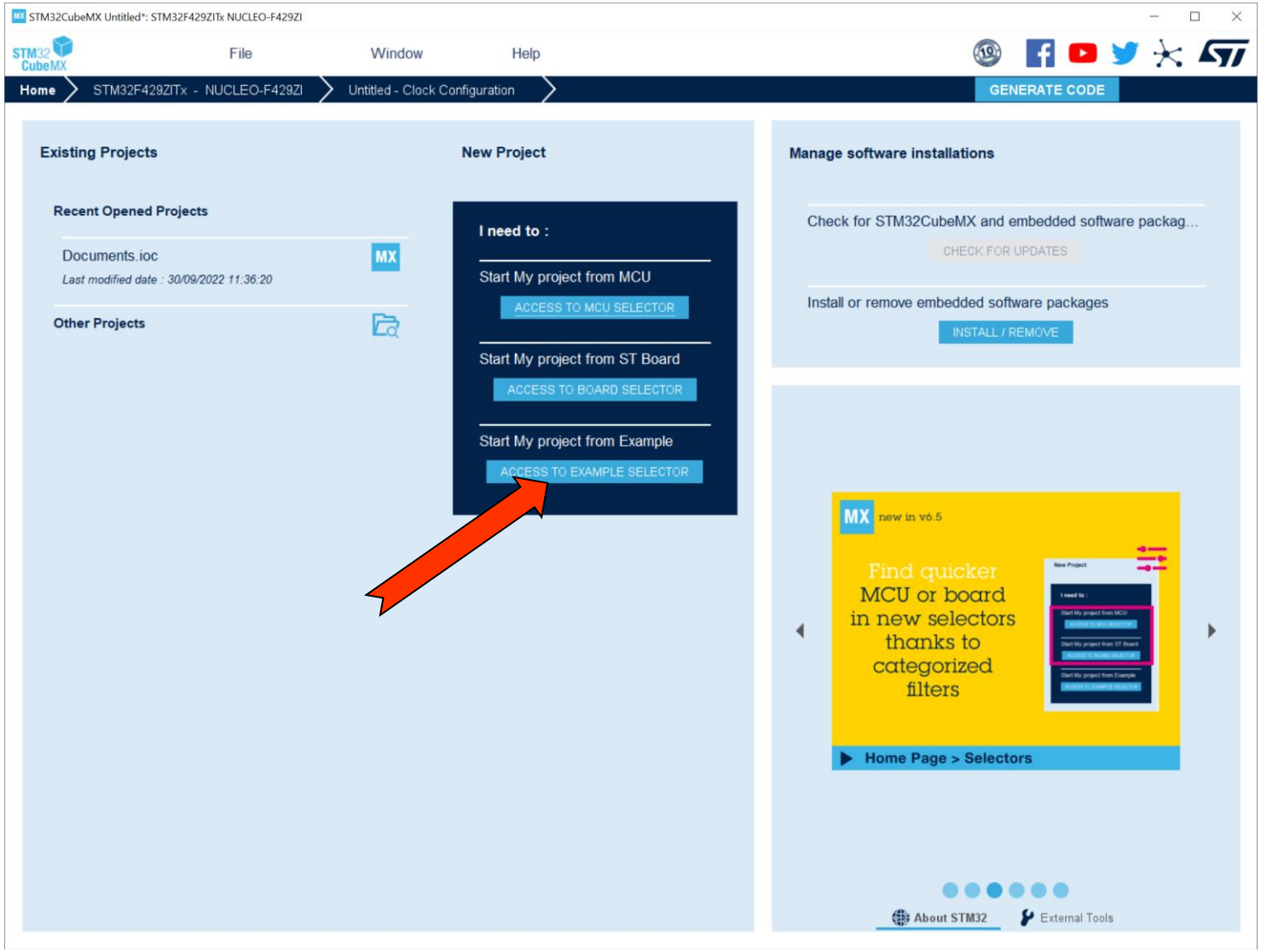
Modos de bajo consumo. Ejemplo aplicación

- Pueden encontrarse ejemplos de aplicación en la Web de ST o en STMCubeMX



Modos de bajo consumo. Ejemplo aplicación

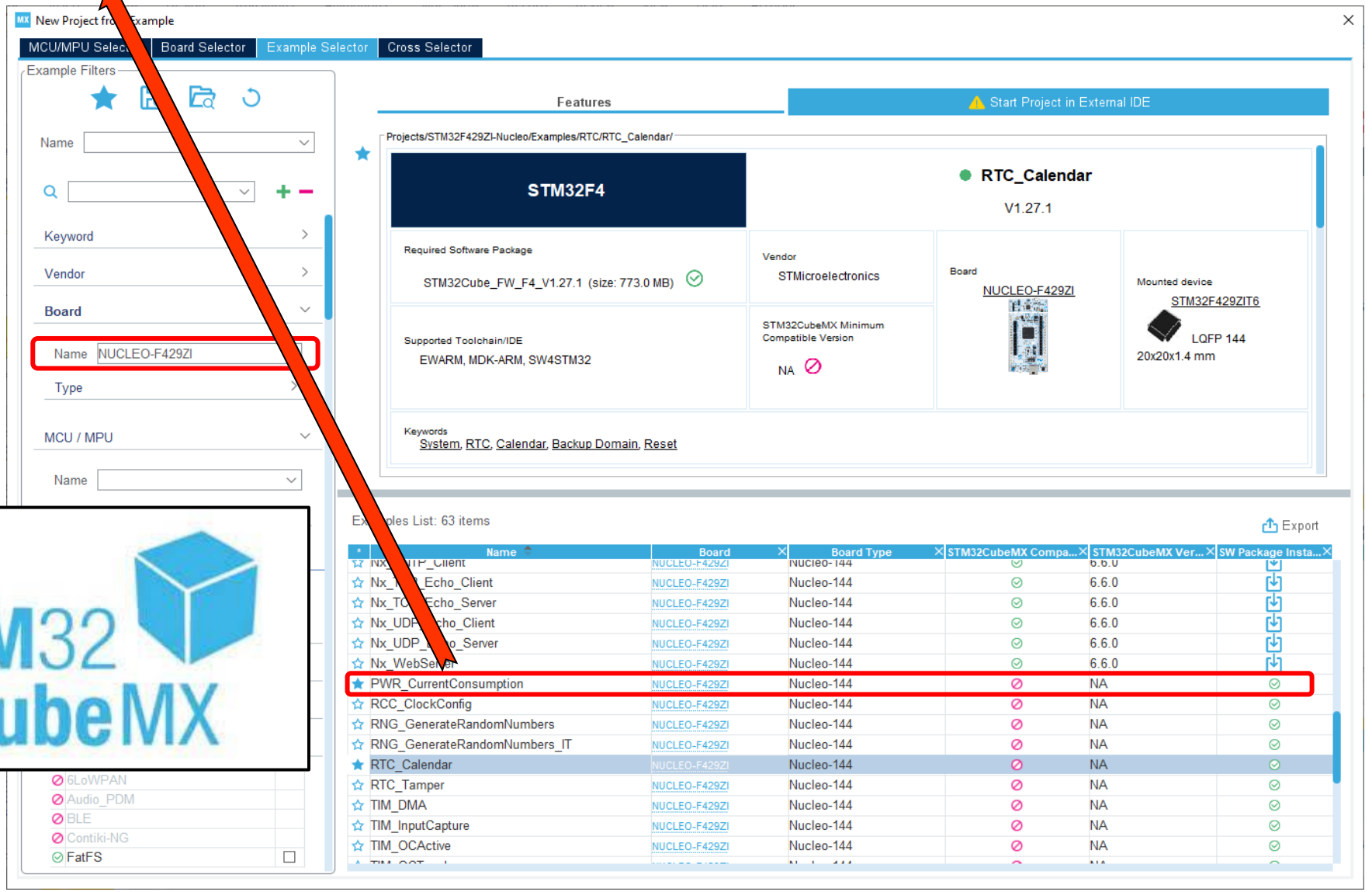
- Pueden encontrarse ejemplos de aplicación en la Web de ST o en STMCubeMX



Modos de bajo consumo. Ejemplo aplicación

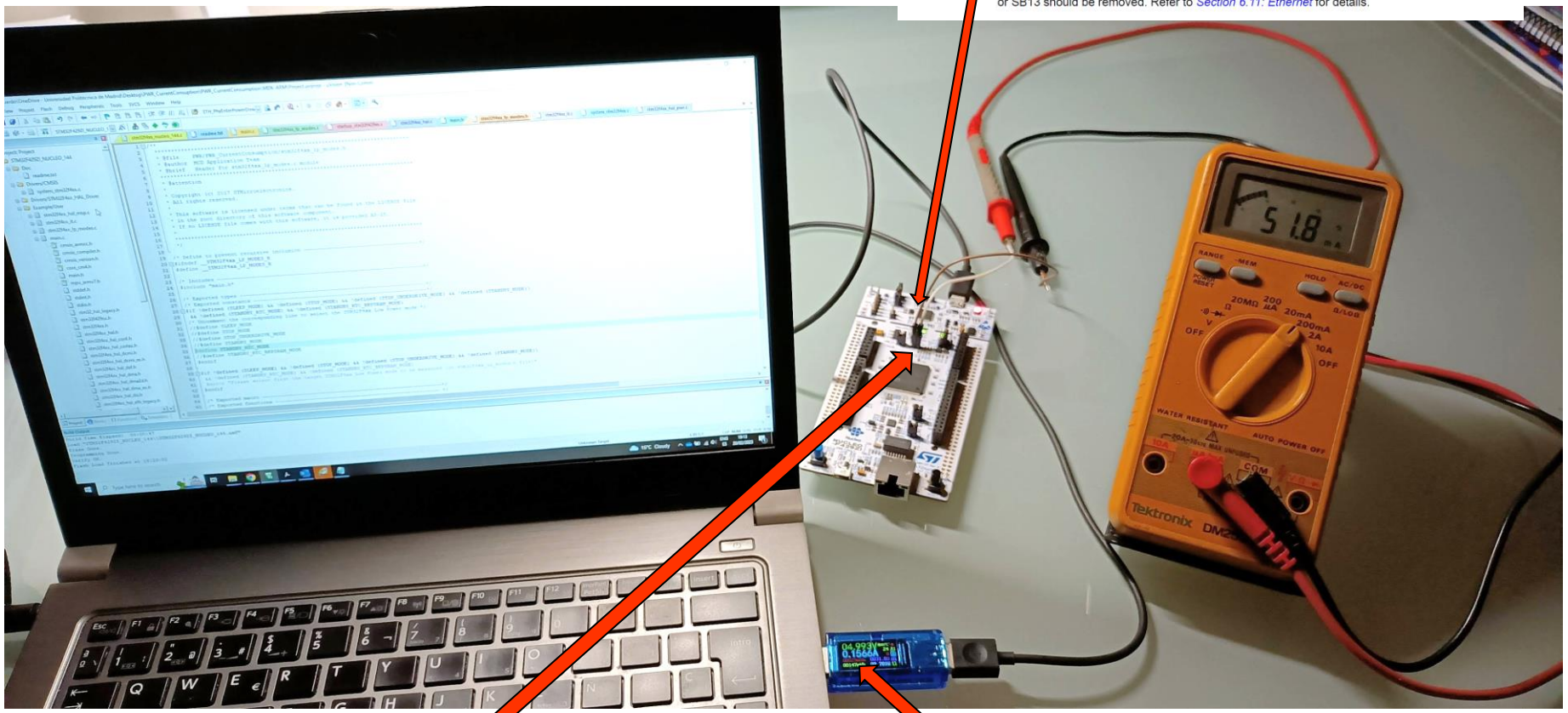
Ejemplo PWR_CurrentConsumption (STM32CubeMX)

Disponible en Moodle



Implementación práctica

- Cómo mido el consumo de mi sistema?



UM1974 Hardware layout and configuration

6.6 Push-buttons

B1 USER: the user button is connected to the I/O PC13 by default (Tamper support, SB173 ON and SB180 OFF) or PA0 (Wakeup support, SB180 ON and SB173 OFF) of the STM32 microcontroller.

B2 RESET: this push-button is connected to NRST and is used to RESET the STM32 microcontroller.

6.7 JP5 (IDD)

Jumper JP5, labeled IDD, is used to measure the STM32 microcontroller consumption by removing the jumper and by connecting an ammeter:

- JP5 ON: STM32 is powered (default)
- JP5 OFF: an ammeter must be connected to measure the STM32 current. If there is no ammeter, the STM32 is not powered

To get a correct current consumption, the Ethernet PHY should be set in power-down mode or SB13 should be removed. Refer to *Section 6.11: Ethernet* for details.

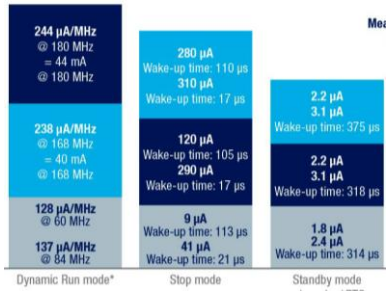
Jumper J5: medida del consumo del micro

El consumo de la placa completa es, evidentemente, mayor

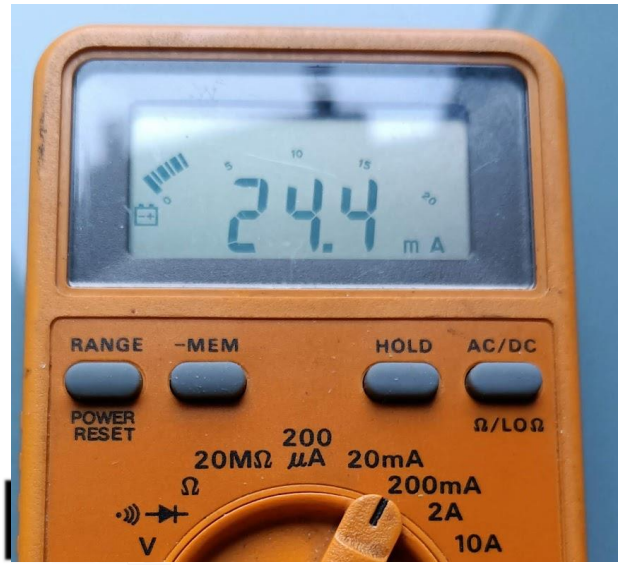
Algunas medidas experimentales

Sistema a 180 MHz modo **RUN**

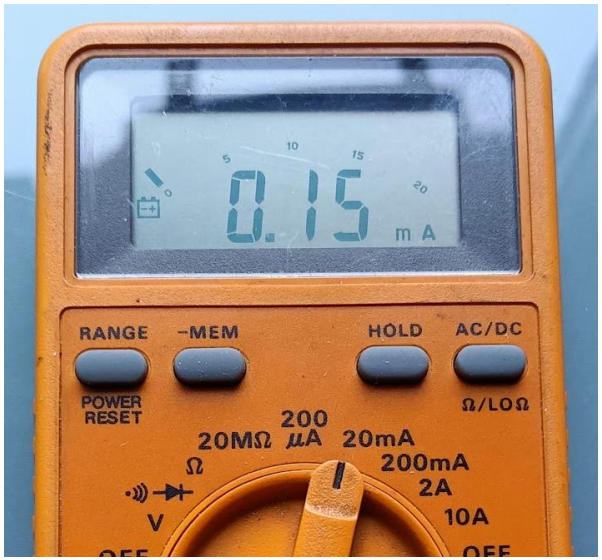
Sistema a 25 MHz modo **RUN**



Modo **SLEEP**



Modo **STOP**



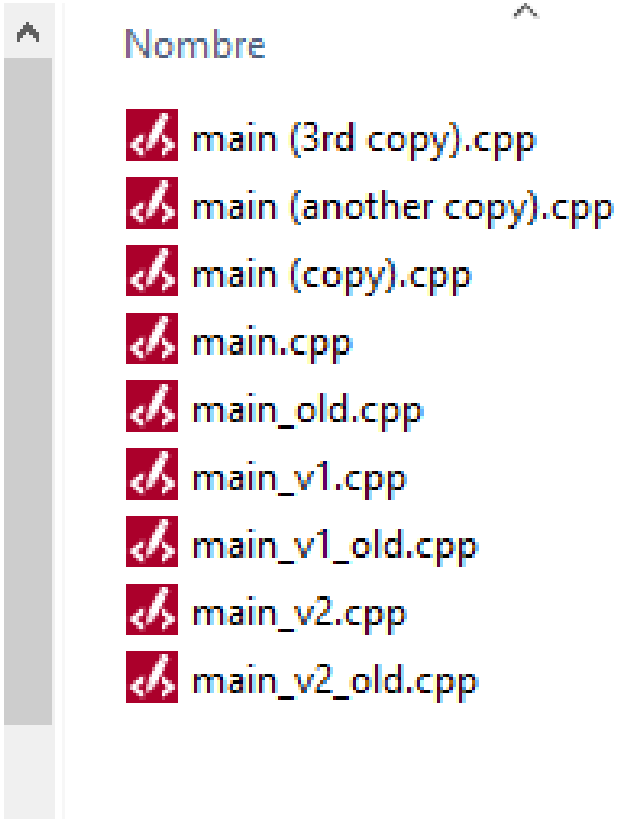
Modo **STANDBY**

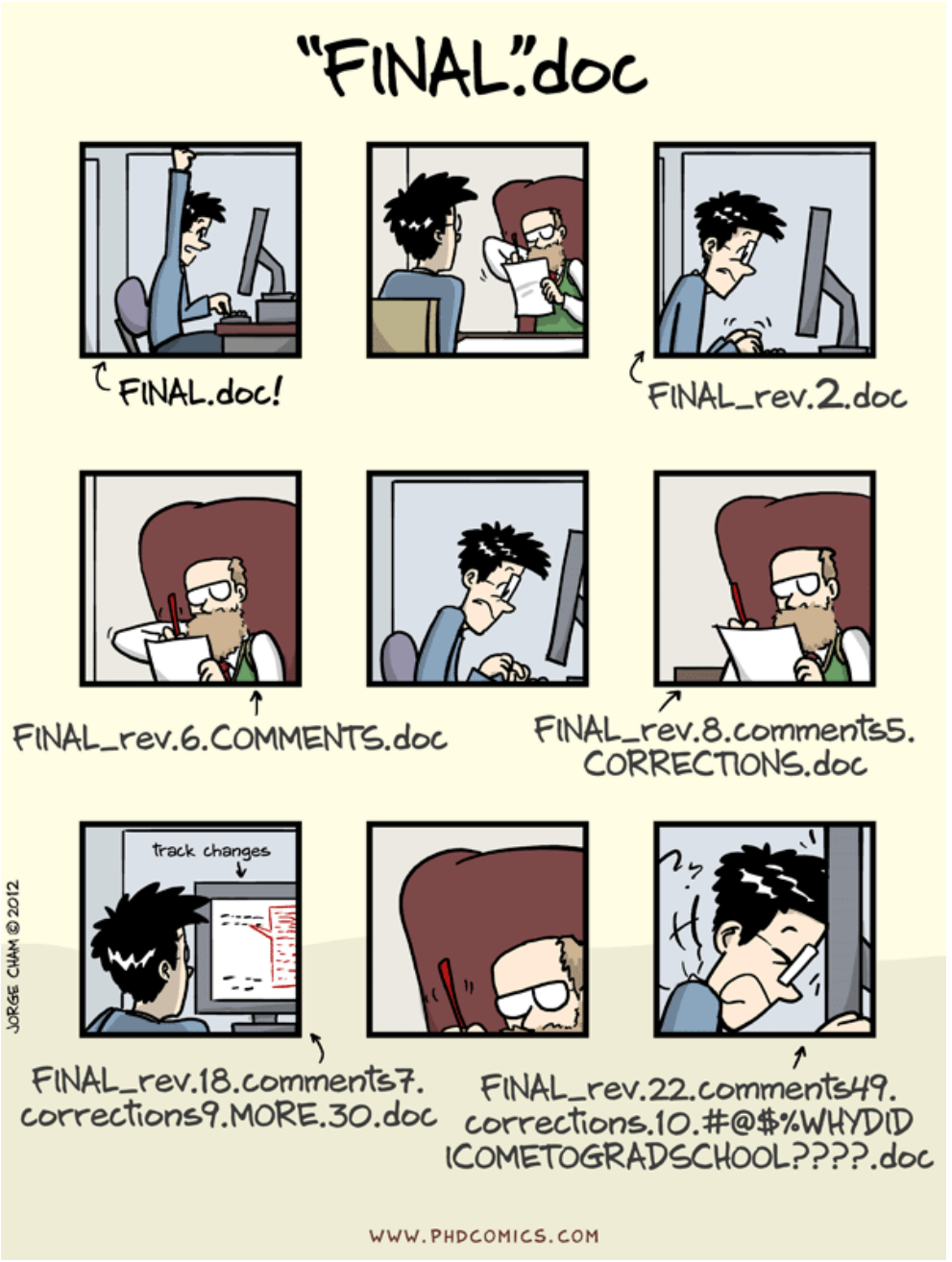
Ingeniería de Sistemas Electrónicos

Programación avanzada de microcontroladores (II)

- Integración de referencia temporal en el sistema
- Desarrollo de aplicaciones "confiables"
- Reducción del consumo
- Desarrollo de software. Control de versiones
- Prácticas de laboratorio

A alguien le suena??





Métodos de seguimiento de versiones

- **Gestión manual de copias**
  - Dificulta organización y versionado de ficheros
    - Complejidad creciente
    - Mucho uso de memoria
    - Fácil cometer errores
    - Difícil compartir
    - Gestión manual, sin herramientas de ayuda
- **Sistemas de almacenamiento online**
  - Dropbox, Owncloud, Onedrive, Google Drive…
  - Pros
    - Sincronización continua
    - Cierta gestión de versiones de ficheros
    - Fácil compartir
  - Cons
    - Trabajo concurrente sobre ficheros
    - Control de las versiones (cuándo?, identificación, seguimiento)
    - Desarrollo de nuevas funcionalidades sobre un código estable

- **Organizar versiones**
  - **Version**
    - "Estado" de un proyecto en un momento dado
  - **Organizar**
    - Historial de cambios
    - Identificar cada versión
    - Recuperar versiones anteriores/diferentes
    - Facilitar la creación de nuevas versiones (con mínimas o múltiples diferencias)
    - Combinar varias versiones
- **Compartir proyectos**
  - Permite acceso a múltiples desarrolladores a un proyecto común
  - Facilita la colaboración
    - Herramienta estándar de gestión de proyectos
    - Seguimiento y trabajo concurrente sobre el mismo o diferentes ficheros

Sistemas de control de versiones: Version Control Systems (VCS)

- Diferentes herramientas de control de versiones disponibles
  - CVS (1986, GNU): Centralized, file-oriented. Deprecated 

  - SVN (2000, Apache): Centralized, project-oriented, heavy branches. 

  - Git (2005, Linus Torvalds): Distributed, project oriented, light branches. 

  - Mercurial (2005, Matt Mackall): Similar to git. Simpler, but less flexible. 

Git

- Sistema de Control de Verciones (VCS) distribuido
- Creado originalmente en 2005
  - Surge de la comunidad de desarrollo de Linux (Linus Torvalds)
  - Diseñado para hacer control de versiones en el kernel de Linux, pero útil en otros entornos
- Características
  - Rápido, sencillo, totalmente distribuido, gran soporte de trabajo concurrente y muchas ramas de desarrollo, adecuado para proyectos muy grandes/pequeños
- Aceptación
  - **MUY usado**
  - ***Recomendación:*** aprended a usar Git, os lo valorarán positivamente y os ahorrará muchos quebraderos de cabeza
  - Estándar de facto actual

Git: documentación

- Sitio web de Git
  - http://git-scm.com/
- Libro online gratuito
  - http://git-scm.com/book
- Tutorial Git
  - http://git-scm.com/docs/gittutorial
- Linea de commandos
  - `git help <command>`

Git hosting

- Existen diferentes servicios con diferentes condiciones

- Los más usados
  - Github
    - https://github.com/

  - GitLab
    - https://about.gitlab.com/

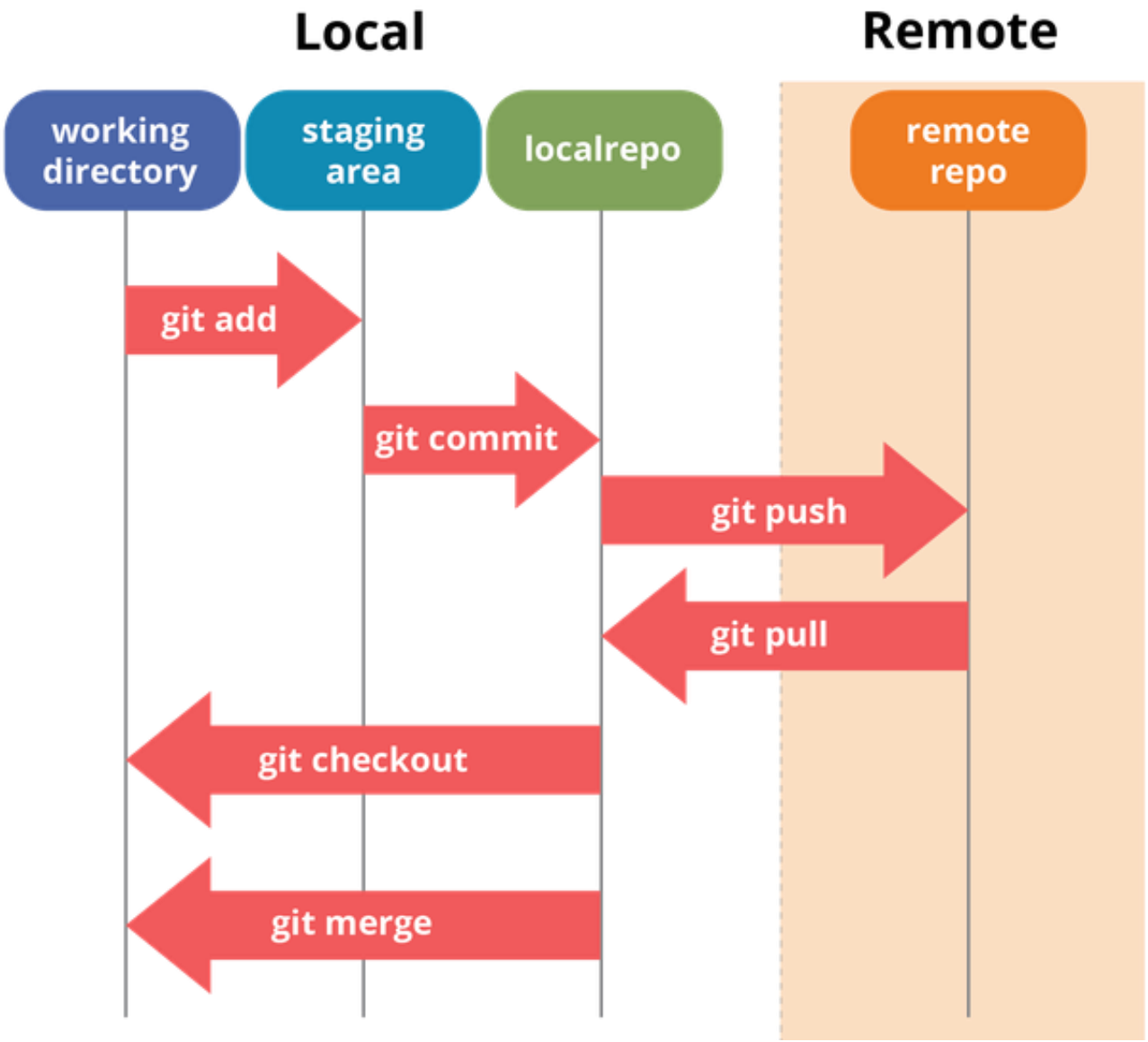
  - Bitbucket
    - https://bitbucket.org/

- Primeros pasos para trabajar con Github:
  - https://guides.github.com/activities/hello-world/
  - https://www.youtube.com/watch?v=HVsySz-h9r4

Git Basic Concepts

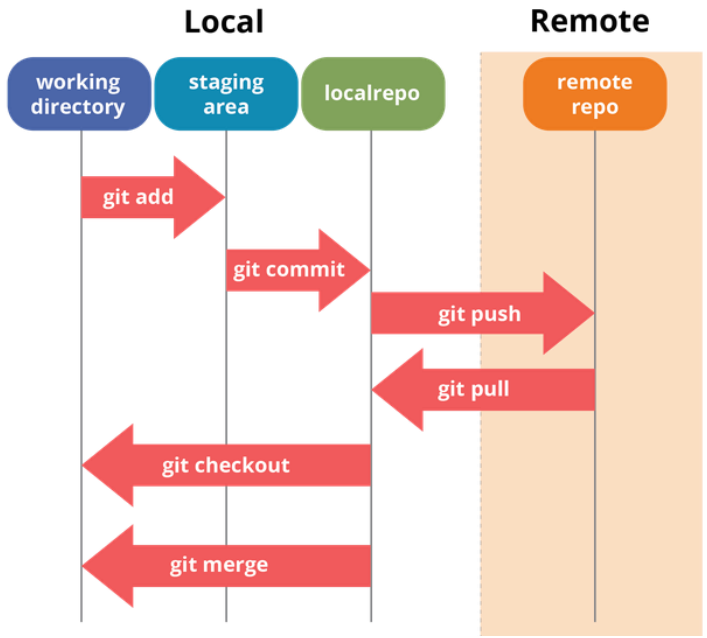
- Repository: Container of the project
  - Contains the different files of the project
  - Contains the multiple versions of each file
  - Normally, there are several repositories for each project
  - Remote repository
    - Normally in a server. Available for all the developers.
  - Local repository
    - Developers start downloading a copy of the remote repository (**clone**)
    - The developer works in their local repository.
    - The developer saves the versions to their local repository (**commit**)
    - When changes are ready to be shared, developers upload their changes to the remote repository (**push**)
    - For obtaining the latest changes in the remote repository, developers can download the latest changes (**pull**)

Áreas Git


Local
Remote
working directory
staging area
localrepo
remote repo
git add
git commit
git push
git pull
git checkout
git merge

Flujo de trabajo básico con Git (en local)

- **1.- Crear/Modificar**
  - Se crean/modifican los ficheros en el working directory
- **2.- Stage**
  - Selecciona ("add") los ficheros a incluir en el siguiente *commit*
  - Añade snapshots de esos ficheros al staging area
    - Únicamente incluye esos ficheros (no otros modificados pero sobre los que no se hace stage)
- **3.- Commit**
  - Almacena **permanentemte** el snapshot con los ficheros del staging area en el directorio .git



Qué es un commit ?

- Un **snapshot** del proyecto
  - Contiene **todos los ficheros** de un proyecto en un momento determinado, no las diferencias con versiones previas
- Un **checkpoint** en el proyecto
  - Puedo volver en cualquier momento
- Cada commit lleva asociado un **commit message**
  - Comentario **descriptivo e inequívoco** del snapshot

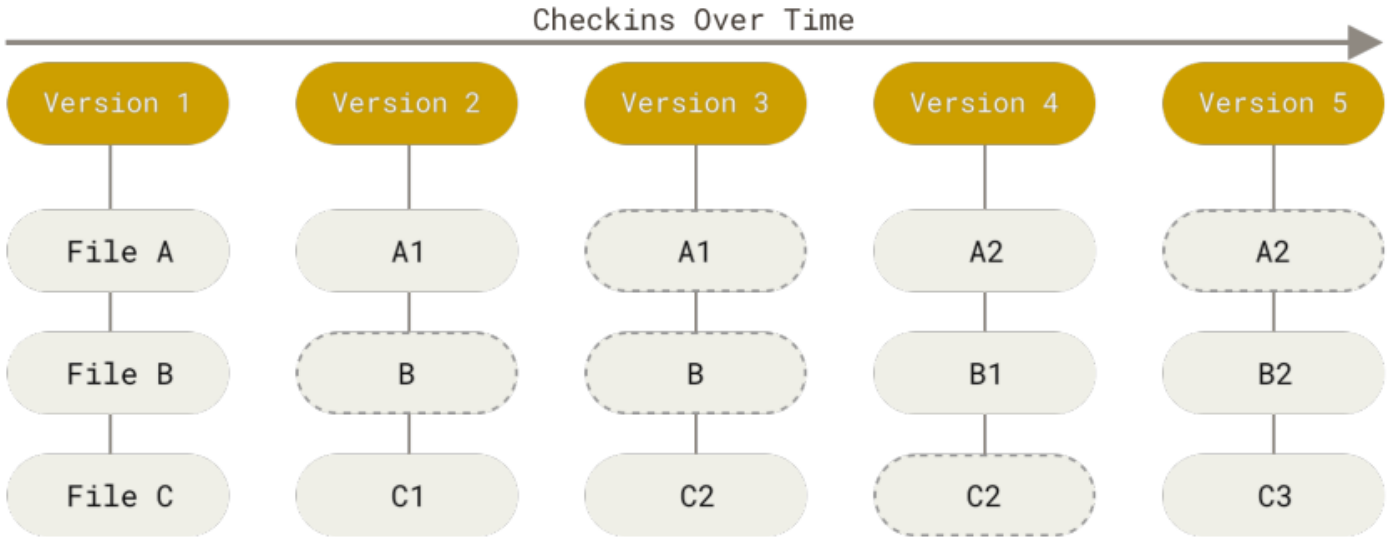


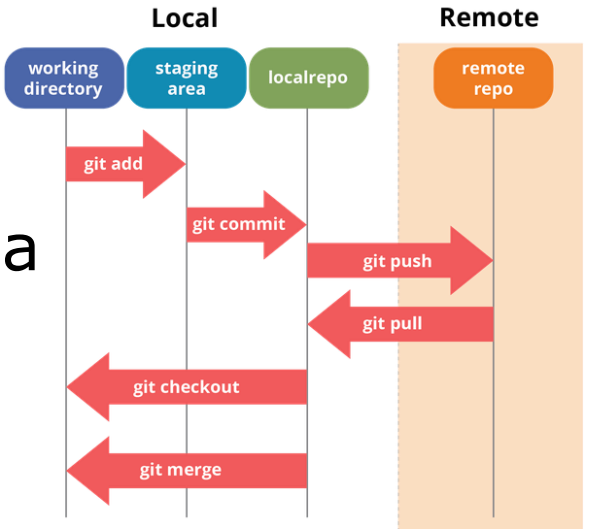
Figure 5. Storing data as snapshots of the project over time

Flujo de trabajo con Git: set-up

- Crear un repositorio en la Web de github
- Clonar el repositorio
  - `git clone` ⟶ Comandos git en terminal (GitBash)
- Añadir ficheros/directorios al repo local
  - copiarlos al modo tradicional
  - `git add` (de los que se quiera hacer commit)
- Primer commit al repo local
  - `git commit –m "Initial commit"`
- Push al repo remoto
  - `git push`
- …Fase de desarrollo en local…

Flujo de trabajo con Git 2: trabajo local + remoto

- Recuperar versión actual del repo remoto (actualiza "Working copy")
  - **`git pull`**
  - `git fetch`
  - `git merge`
- Modificar ficheros (del "Working copy" o copia local)
- Revisar cambios (respecto del repositorio)
  - `git status`
  - `git diff`
  - `git log`
- Añadir ficheros modificados al stage area
  - **`git add`**
- Commit cambios
  - **`git commit`**
- Incorporar los cambios en el repo remoto
  - **`git push`**



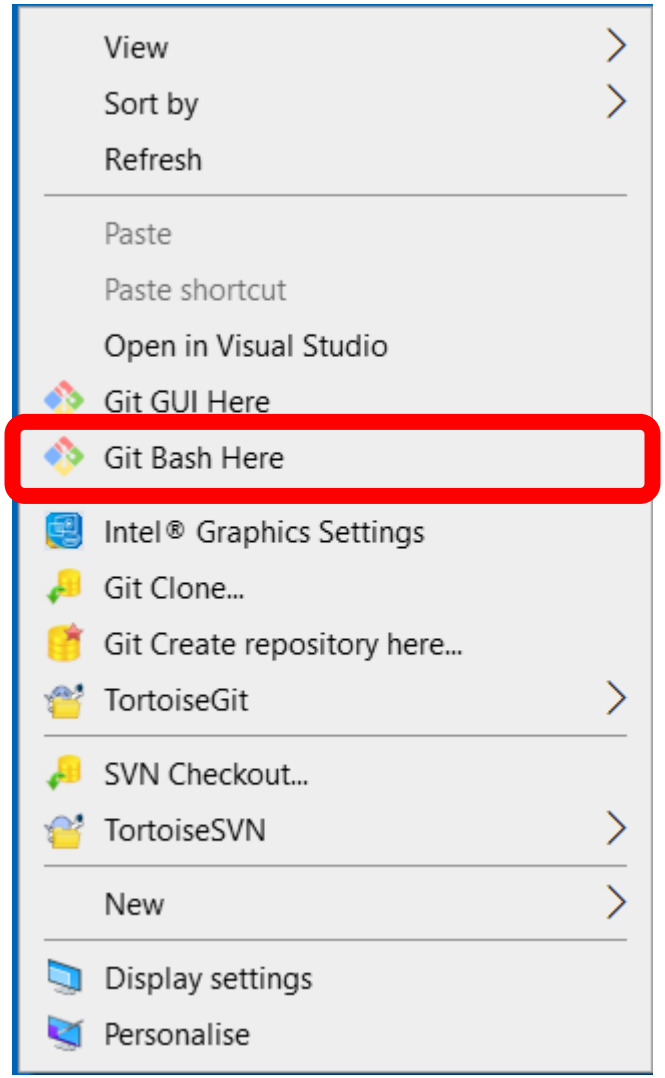
Using GIT with Keil environment

https://www.keil.com/appnotes/files/apnt_279.pdf

- ***Project Files under Version Control***
  - Before setting up the workflow, the project manager should be absolutely clear about the files that need to be version controlled. Of course all source code files need to be versioned, but there are a couple of files that are special to µVision that need to be monitored as well.
  - All user generated source files (*.c, *.cpp, *.h, *.inc, *.s)
  - Project file: *Project*.uvprojx (is used to build the project from scratch)
  - Project options file: *Project*.uvoptx (contains information about the debugger and trace configuration)
  - Configuration files for the run-time environment that are copied to the project (all files below .\RTE)
  - List of #includes created by software components: RTE\RTE_Components.h file
  - Device configuration file: for example RTE\Device\LPC1857\RTE_Device.h
  - Linker control file (*Project*.sct) if created manually
  - All relevant Pack files (for example ARM::CMSIS, Keil::Middleware, Device Family Packs, etc.)
- ***Files that do not need to be monitored***
  - Project screen layout file: *Project*.uvguix.*username*
  - All files that are part of a Pack (the complete Pack will be revision controlled and is available to every user as soon as he is installing it using Pack Installer)
  - Generated output files in the sub-directories .\Listings and .\Objects
  - INI files for debug adapters

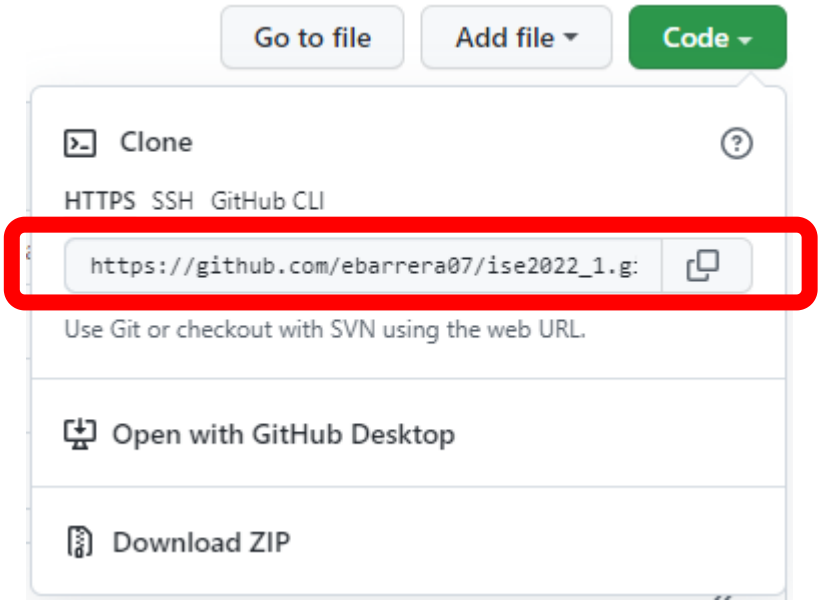
Steps to create your first repository

1. Access to github.com
2. Create an account
3. Sign-in
4. Create a new repository
5. Install git for your operating system
   - Windows(https://git-scm.com/download/win)
6. In the windows explorer, select the folder where you want to have your repository, right click and select Git Bash Here



Steps to create your first repository

7. In the online github page navigate to your repository an click on Code button
8. Using the command window execute
   - git clone <complete repo url>
9. After some seconds you will have a copy of your project in a local folder.
10. Start working with your code an learn to use the commands *git add, git commit* and *git push* (*pull*)

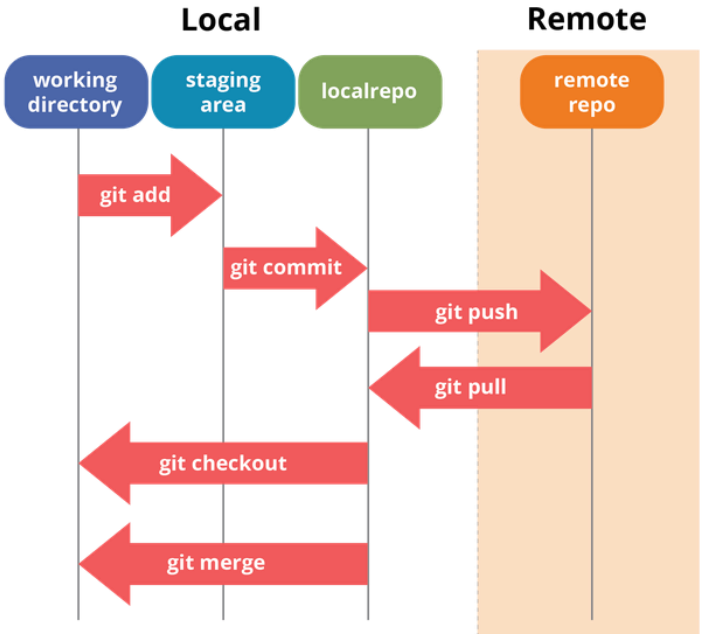


Pasos para trabajar con GIT en ISE

- Crear cuenta en github con correo electrónico de alumnos UPM
- Crear un repositorio para cada estudiante
  - Nombre del repo: ISE2025_grupoLAB_ApellidoNombre
- Añada como colaborador a los profesores de la asignatura (eduardo.barrera@upm, julian.nieto.valhondo@upm.es)
- Crear carpeta/s local/es para alojar el/los repositorio/s local/es (laboratorio, portátil, casa)
- Abrir Git Bash (click derecho) en dicha carpeta
- Configurar entorno -usuario y email- (solo una vez)
  - `git config user.name "Mona Lisa“`
  - `git config user.email "email@example.com"`

Pasos para trabajar con GIT en ISE

- Clonar el repositorio en la carpeta local (solo una vez)
  - `git clone <repository_url>`
- Copiar el ejemplo del servidor Web original (sin modificar), tras hacer un clean del proyecto
- Añadir los ficheros de los que se desee hacer commit (puede hacerse uso de `.gitignore`)
  - `git add <path>`
- Hacer commit (con comentarios) del ejemplo original
  - `git commit –m “comentarios”`
- Hacer push para subir el repositorio local a remote
  - `git push`



Pasos para trabajar con GIT en ISE

- Revisar el repositorio remoto en la Web de github
- Añadir las modificaciones al ejemplo del servidor en el repositorio local
- Añadir los ficheros de los que se desee hacer commit (puede hacerse uso de `.gitignore`)
  - `git add <path>`
- Hacer uno o varios commit (con comentarios) de las modificaciones que se han realizado
  - `git commit –m “comentarios”`
- Hacer push para subir el repositorio local a remote
  - `git push`
- Revisar nuevamente el repositorio remoto en la Web de github

Pasos para trabajar con GIT en ISE

- Todo el código desarrollado en la asignatura deberá estar en github
- En cada sesion de laboratorio se harán varios commit (con comentarios descriptivos de los cambios) y al menos un push al finalizar la sesión
- En el trabajo no presencial deberán realizarse igulamente commit con comentarios y al menos un push por cada día que se trabaje en la asignatura
- Cuando comience el Bloque 2 de la asignatura se deberá crear un repositorio por cada Equipo de Trabajo
- Añada como colaborador a su profesor de la asignatura

Ingeniería de Sistemas Electrónicos

Programación avanzada de microcontroladores (II)

- Integración de referencia temporal en el sistema
- Desarrollo de aplicaciones "confiables"
- Reducción del consumo
- Desarrollo de software. Control de versiones
- Prácticas de laboratorio

- Preliminar: Integración de Práctica 1 en sistema de Control de Versiones (GIT) (libre acceso)
- **Práctica 2**: Integración de SNTP y RTC en servidor
  - 1 semana (usar repo Git individual)
- **Práctica 3**: Modos de bajo consumo y Watchdog
  - 1 semana (usar repo Git individual)

- **Entrega P1, P2 y P3** en Moodle → 28 feb, 7 mar, 14 mar (solo proyectos Keil)
- Entrega **documentación** completa del **Bloque 1** → viernes, **21 de marzo**
- **Evaluación individual Bloque 1** → lunes, **31 de marzo**

Desarrollo de la asignatura

- Entregas Bloque 1

	L	M	X	J	V	S	D
Feb	3	4	5	6	7	8	9
	10	11	12	13	14	15	16
	17	18	19	20	21	22	23
Mar	24	25	26	27	28	1	2
	3	4	5	6	7	8	9
	10	11	12	13	14	15	16
	17	18	19	20	21	22	23
	24	25	26	27	28	29	30
Abr	31	1	2	3	4	5	6
	7	8	9	10	11	12	13
	14	15	16	17	18	19	20
	21	22	23	24	25	26	27
May	28	29	30	1	2	3	4
	5	6	7	8	9	10	11
	12	Hor J.13	14	15	16	17	18
	19	20	21	22	23	24	25
	26	27	28	29	30	31	1
Jun	30	1	2	3	4	5	6

PROPUESTAS DISEÑO BLOQUE 2

Lunes 17 febrero:

- Equipos de 3/4 estudiantes (del mismo grupo de laboratorio)
- Representante de cada Equipo

Lunes 3 de marzo:

- Propuestas de diseño Bloque 2

Lunes 17 de marzo:

- Presentaciones propuestas Bloque 2

Prácticas BLOQUE 1

Viernes 28 de febrero:

- Práctica 1

Viernes 7 de marzo:

- Práctica 2

Viernes 14 de marzo:

- Práctica 3

Viernes 21 de marzo:

- Documentación Bloque 1