



Timers and STM32

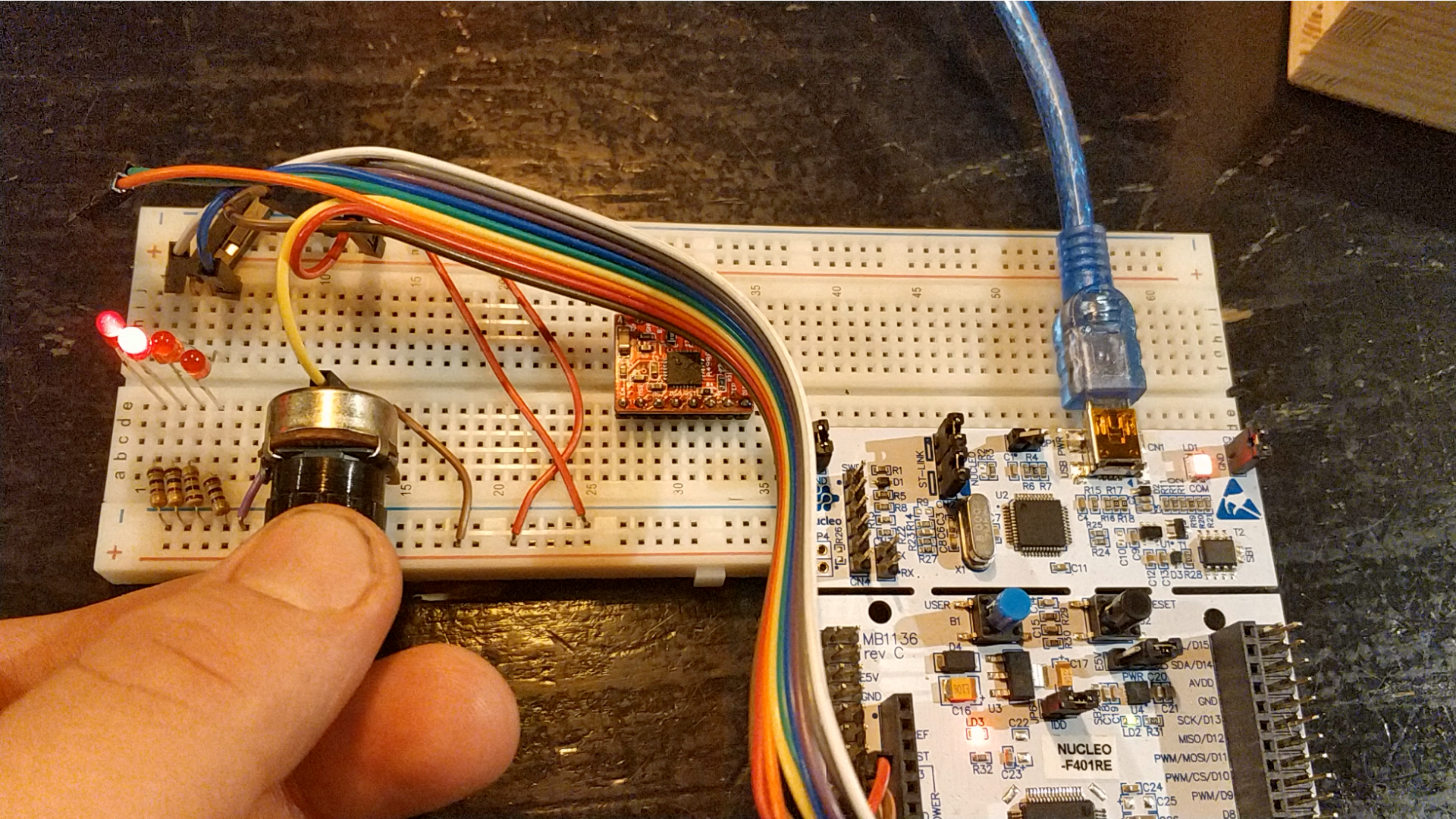
How to learn fast and stay sane



PIOTR SUWAŁA / REGULAR JAVA DEVELOPER

HIDDEN AGENDA

- **STM32/GENERIC TOOLING**
- **STM32 DOCUMENTATION**
- **LEARNING TECHNIQUES**
 - **MIND MAPS**
 - **ORGANISED LEARNING**
- **TIMERS/PWM**
 - **DEBUGGING**
- **DMA/ADC**
 - **DEBUGGING**






```
101  int main(void) {  
102     HAL_Init();  
103     SystemClock_Config();  
104     MX_GPIO_Init();  
105     MX_TIM3_Init();  
106     MX_TIM4_Init();  
107     MX_TIM5_Init();  
108     MX_TIM2_Init();  
109     DMA_INIT();  
110     ADC1_INIT();  
111  
112     // No code in main loop  
113     while (1);  
114      }
```


Preparing

Tools, documentation, organising.





Existing Projects

Recent Opened Projects

PWMFROMSTM.ioc

MX

Last modified date : 18/11/2019 21...

PWM.ioc

MX

Last modified date : 11/11/2019 15...

Other Projects



New Project

I need to :

Start My project from MCU

ACCESS TO MCU SELECTOR

Start My project from STBoard

ACCESS TO BOARD SELECTOR

Start My project from Cross ...

ACCESS TO CROSS SELECTOR

Manage software installations

Check for STM32CubeMX an...

CHECK FOR UPDATES

Install or remove embedded ...

INSTALL / REMOVE



STM32CubeMX

FileWindowHelp

10

f

▶

ST

HomeSTM32F401RETx - NUCLEO-F401REUntitled - Project ManagerGENERATE CODE

Pinout & ConfigurationClock ConfigurationProject ManagerTools

Project

Code Generator

Advanced Settings

Project Settings

Project Name

PWMFROMSTM

Project Location

/Users/psuwala/projects/STM32/

Browse

Application Structure

Basic

Do not generate the ma...

Toolchain Folder Location

/Users/psuwala/projects/STM32/PWMFROMSTM/

Toolchain / IDE

SW4STM32

Generate Under Root

Linker Settings

Minimum Heap Size

0x200

Minimum Stack Size

0x400

Mcu and Firmware Package

Mcu Reference

STM32F401RETx

Firmware Package Name and Version

STM32Cube FW_F4 V1.24.1

Use latest available version

Cmake based IDE

PWMFROMSTM ~/projects/STM32/PWMFROMSTM

cmake-build-debug

CMakeFiles

Drivers

CMSIS

STM32F4xx_HAL_Driver

Inc

main.h

stm32f4xx_hal_conf.h

stm32f4xx_it.h

Src

main.c

stm32f4xx_hal_msp.c

stm32f4xx_it.c

syscalls.c

system_stm32f4xx.c

startup

startup_stm32f401xe.s

.cproject

.gdbinit

.gitignore

.mxproject

.osx.project

.project

cmake_install.cmake

CMakeCache.txt

CMakeLists.txt

CMakeLists_template.txt

Makefile

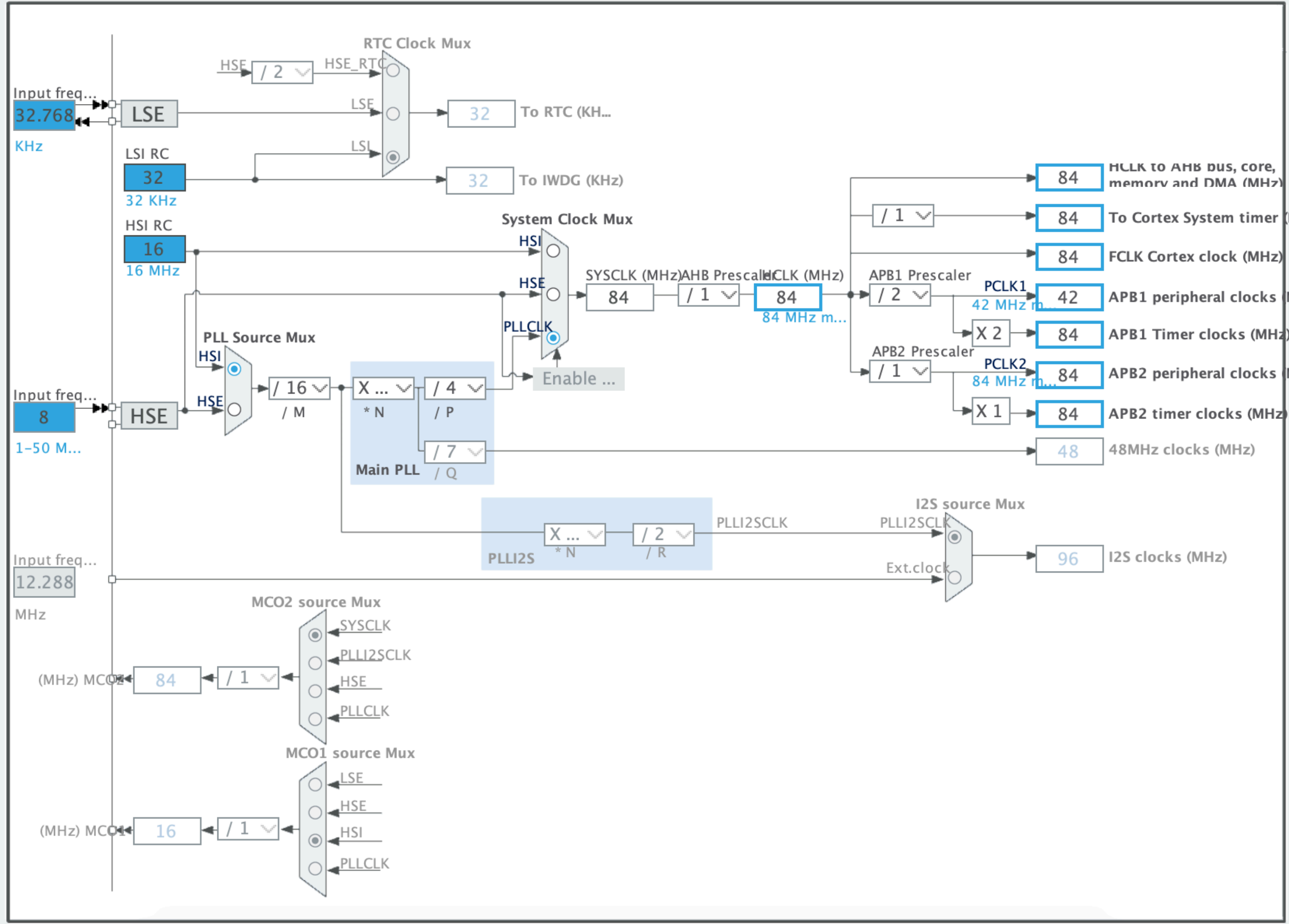
PWMFROMSTM.bin

PWMFROMSTM.hex

PWMFROMSTM.ioc

PWMFROMSTM.map

STM32F401RETx_FLASH.ld



TIM2 Mode and Configuration

Mode

Slave Mode	Disable
Trigger Source	Disable
Clock Source	Internal Clock
Channel1	PWM Generation CH1
Channel2	Disable
Channel3	Disable
Channel4	Disable
Combined Channels	Disable

Configuration

Reset Configuration

✓ NVIC Settings

✓ DMA Settings

✓ GPIO Settings

✓ Parameter Settings

✓ User Constants

Configure the below parameters :

Search (Ctrl+F)

Counter Settings

Prescaler (PSC – 16 bits value) 8400

Counter Mode Up

Counter Period (AutoReload ...) 10000

Internal Clock Division (CKD) No Division

auto-reload preload Enable

Trigger Output (TRGO) Parameters

Master/Slave Mode (MSM bit) Disable (Trigger input effect not delay...

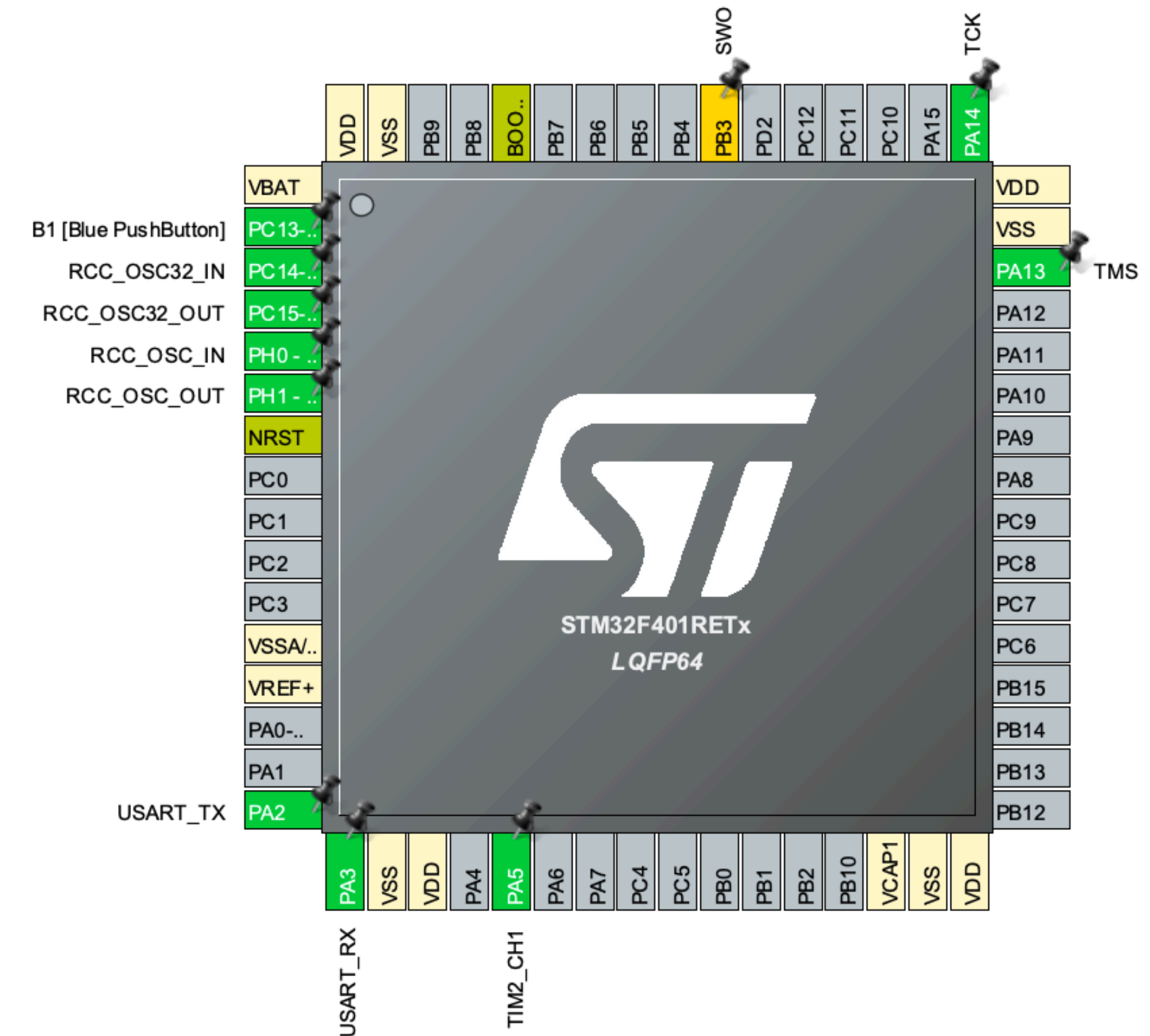
Trigger Event Selection Update Event

PWM Generation Channel 1

Mode PWM mode 1

Pinout view

System view




```

/**
 * @brief System Clock Configuration
 * @retval None
 */
void SystemClock_Config(void) {
    RCC_OscInitTypeDef RCC_OscInitStruct = { .OscillatorType: 0};
    RCC_ClkInitTypeDef RCC_ClkInitStruct = { .ClockType: 0};

    /** Configure the main internal regulator output voltage
    */
    __HAL_RCC_PWR_CLK_ENABLE();
    __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE2);
    /** Initializes the CPU, AHB and APB busses clocks
    */
    RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
    RCC_OscInitStruct.HSIState = RCC_HSI_ON;
    RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
    RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
    RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI;
    RCC_OscInitStruct.PLL.PLLM = 16;
    RCC_OscInitStruct.PLL.PLLN = 336;
    RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV4;
    RCC_OscInitStruct.PLL.PLLQ = 7;
    if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK) {
        Error_Handler();
    }
    /** Initializes the CPU, AHB and APB busses clocks
    */
    RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK | RCC_CLOCKTYPE_SYSCLK
        | RCC_CLOCKTYPE_PCLK1 | RCC_CLOCKTYPE_PCLK2;
    RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
    RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
    RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;
    RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

    if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK) {
        Error_Handler();
    }
}

```


The image shows the CLion IDE interface for a project named PWMFROMSTM. The main editor window displays the main.c file, which contains C code for system clock configuration. The code includes comments in Italian and defines two structures, RCC_OscInitStruct and RCC_ClkInitStruct, to configure the system clock. The code also includes a main function that calls the SystemClock_Config function. The left sidebar shows the project structure, including the PWMFROMSTM directory, its subdirectories (cmake-build-debug, CMakeFiles, Drivers, Inc, Src, startup), and various files (main.h, stm32f4xx_hal_conf.h, stm32f4xx_it.h, main.c, stm32f4xx_hal_msp.c, stm32f4xx_it.c, syscalls.c, system_stm32f4xx.c, startup_stm32f401xe.s, .cproject, .gdbinit, .gitignore, .mxproject, .osx.project, .project, cmake_install.cmake, CMakeCache.txt, CMakeLists.txt, CMakeLists_template.txt, Makefile, PWMFROMSTM.bin, PWMFROMSTM.hex, PWMFROMSTM.ioc, PWMFROMSTM.map, STM32F401RETx_FLASH.ld). The bottom status bar shows the current file is main.c and the project is in the Debug configuration.


```
GPIO_InitStruct.Pin = GPIO_PIN_1;  
GPIO_InitStruct.Mode = GPIO_MODE_ANALOG;  
GPIO_InitStruct.Pull = GPIO_NOPULL;  
GPIO_InitStruct.
```

HAL_GPIO_I	F	🔒		uint32_t
	F	🔒	Alternate	uint32_t
	F	🔒	Mode	uint32_t
	F	🔒	Pin	uint32_t
	F	🔒	Pull	uint32_t
	F	🔒	Speed	uint32_t

Press ^⇧Space to filter results by type [Next Tip](#)



```
/* USER CODE END 4 */
```


How do I apply?

There are a few ways you can apply for a free license. The best way is to use your official institutional email address or ISIC card. In this case, it will only take you a few minutes to get a free educational license pack.

You can also apply with a student/teacher card or any other official document certifying your affiliation with your academic institution. Such applications may take several days to process.

Before applying, make sure to read the [License Terms](#) carefully.


```
MacBook-Pro-Piotr :: projects/STM32/PWMFROMSTM <master*> » sudo openocd -f interface/stlink-v2-1.cfg -f target/stm32f4x.cfg -c init -c "reset init"
Open On-Chip Debugger 0.10.0
Licensed under GNU GPL v2
[For bug reports, read
[      http://openocd.org/doc/doxygen/bugs.html
[Info : auto-selecting first available session transport "hla_swd". To override use 'transport select <transport>'.
Info : The selected transport took over low-level target control. The results might differ compared to plain JTAG/SWD
adapter speed: 2000 kHz
adapter_nsrst_delay: 100
none separate
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
Info : clock speed 1800 kHz
Info : STLINK v2 JTAG v29 API v2 SWIM v18 VID 0x0483 PID 0x374B
Info : using stlink api v2
Info : Target voltage: 3.276467
Info : stm32f4x.cpu: hardware has 6 breakpoints, 4 watchpoints
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
adapter speed: 1800 kHz
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x08001970 msp: 0x20018000
Info : Unable to match requested speed 8000 kHz, using 4000 kHz
Info : Unable to match requested speed 8000 kHz, using 4000 kHz
adapter speed: 4000 kHz
Info : accepting 'gdb' connection on tcp/3333
Info : device id = 0x10016433
Info : flash size = 512kbytes
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
adapter speed: 1800 kHz
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x08001970 msp: 0x20018000
Info : Unable to match requested speed 8000 kHz, using 4000 kHz
Info : Unable to match requested speed 8000 kHz, using 4000 kHz
adapter speed: 4000 kHz
target halted due to breakpoint, current mode: Thread
xPSR: 0x61000000 pc: 0x20000046 msp: 0x20018000
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
adapter speed: 1800 kHz
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x08001970 msp: 0x20018000
```



```
[MacBook-Pro-Piotr :: projects/STM32/PWMFROMSTM <master*> » cat .gdbinit
tar remote :3333
load PWMFROMSTM.elf
file PWMFROMSTM.elf
[MacBook-Pro-Piotr :: projects/STM32/PWMFROMSTM <master*> » arm-none-eabi-gdb
GNU gdb (GNU Tools for Arm Embedded Processors 8-2018-q4-major) 8.2.50.20181213-git
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=x86_64-apple-darwin10 --target=arm-none-eabi".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".
warning: No executable has been specified and target does not support
determining executable automatically. Try using the "file" command.
0x08001970 in ?? ()
Loading section .isr_vector, size 0x194 lma 0x8000000
Loading section .text, size 0x18a0 lma 0x8000194
Loading section .rodata, size 0x10 lma 0x8001a34
Loading section .ARM, size 0x8 lma 0x8001a44
Loading section .init_array, size 0x4 lma 0x8001a4c
Loading section .fini_array, size 0x4 lma 0x8001a50
Loading section .data, size 0xc lma 0x8001a54
Start address 0x8001970, load size 6752
Transfer rate: 11 KB/sec, 964 bytes/write.
(gdb) █
```

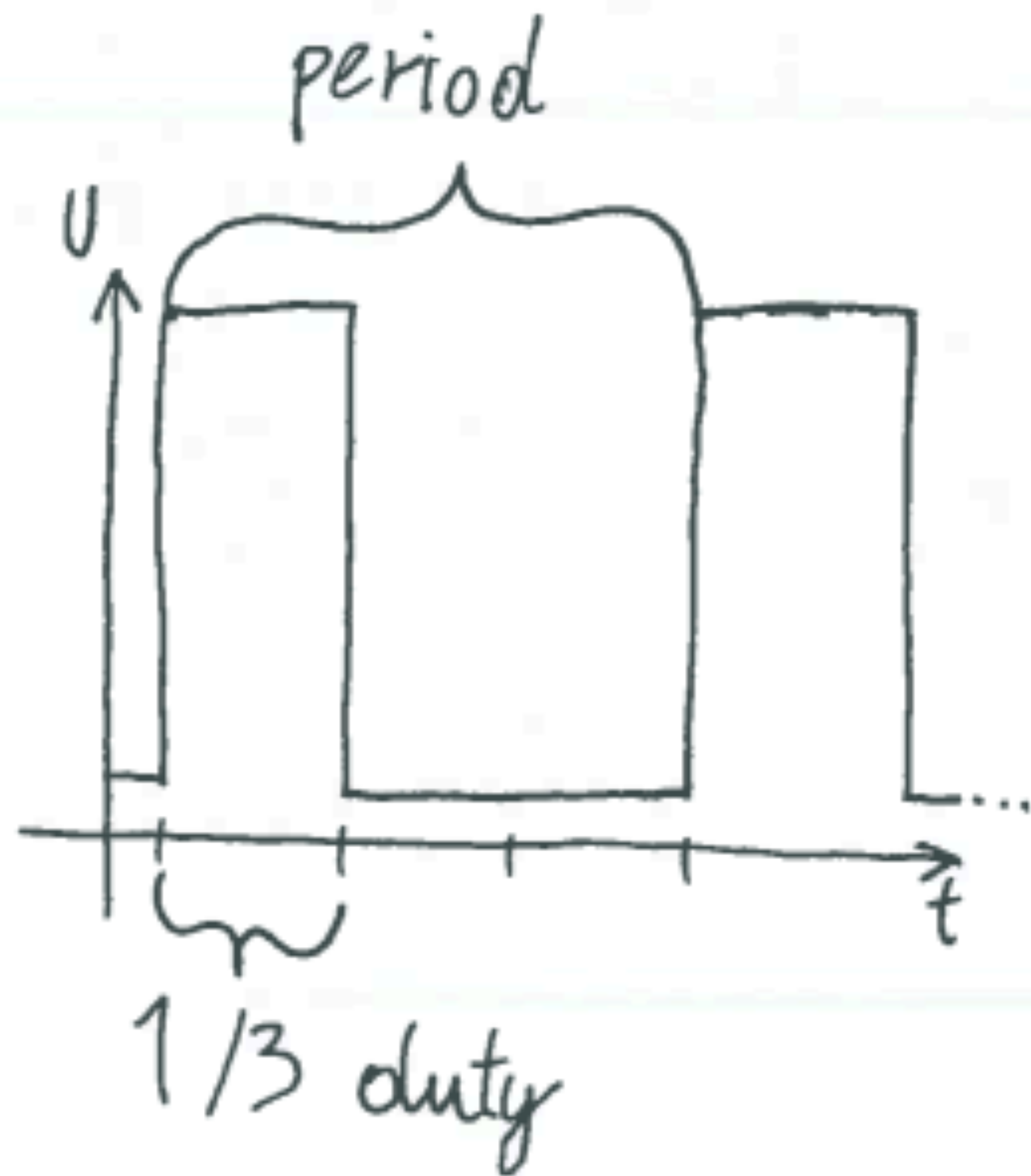



Fucking
Timers

How do
they work
?



Pulse Width Modulation



13	General-purpose timers (TIM2 to TIM5)	316
13.1	TIM2 to TIM5 introduction	316
13.2	TIM2 to TIM5 main features	316
13.3	TIM2 to TIM5 functional description	317
13.3.1	Time-base unit	317
13.3.2	Counter modes	319
13.3.3	Clock selection	328
13.3.4	Capture/compare channels	331
13.3.5	Input capture mode	333
13.3.6	PWM input mode	334
13.3.7	Forced output mode	335
13.3.8	Output compare mode	335
13.3.9	PWM mode	336
13.3.10	One-pulse mode	339
13.3.11	Clearing the OCxREF signal on an external event	340
13.3.12	Encoder interface mode	341
13.3.13	Timer input XOR function	344
13.3.14	Timers and external trigger synchronization	344
13.3.15	Timer synchronization	347
13.3.16	Debug mode	352

PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to [Upcounting mode](#).

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxREF is held at '0'. [Figure 117](#) shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

Figure 117. Edge-aligned PWM waveforms (ARR=8)

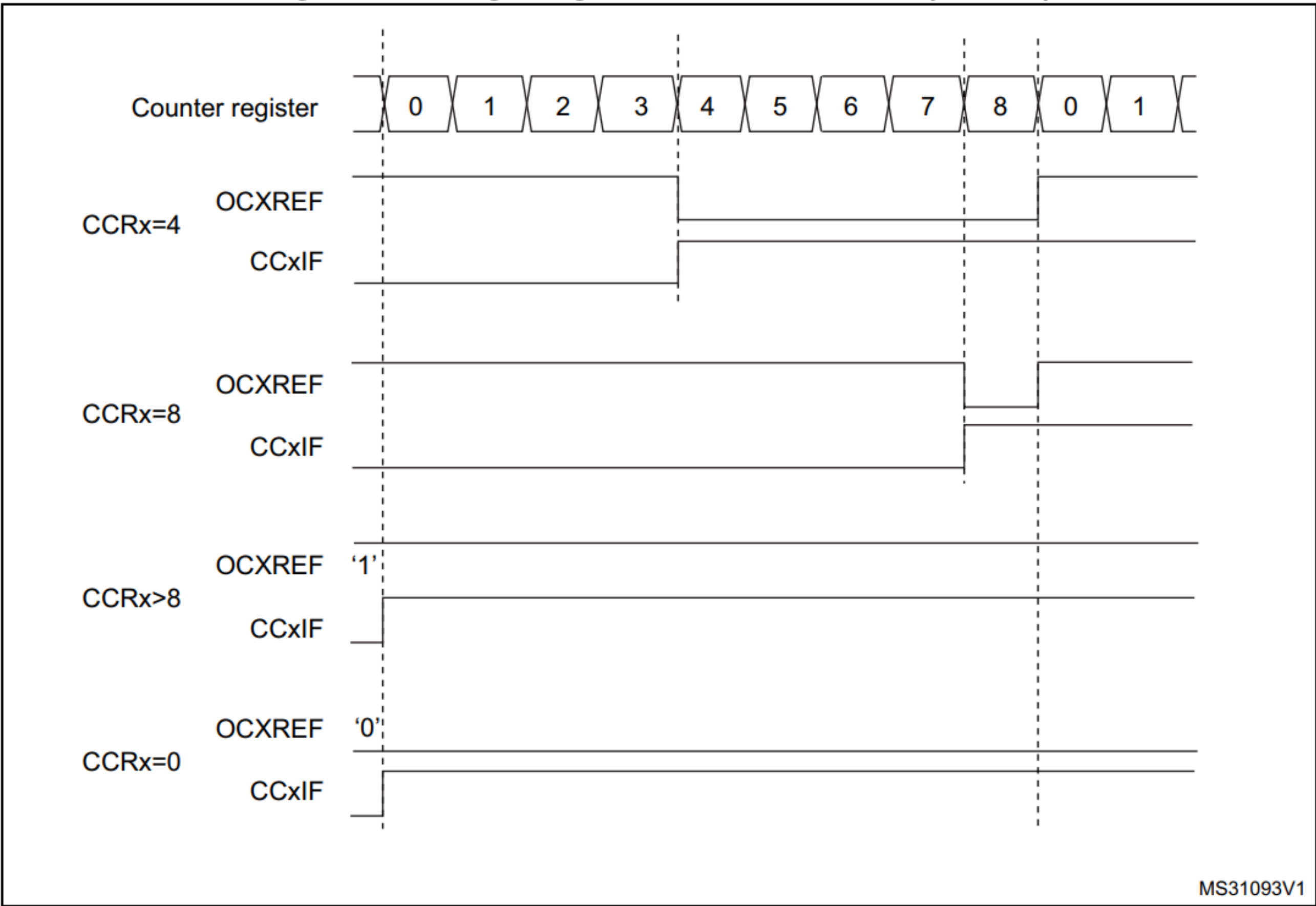
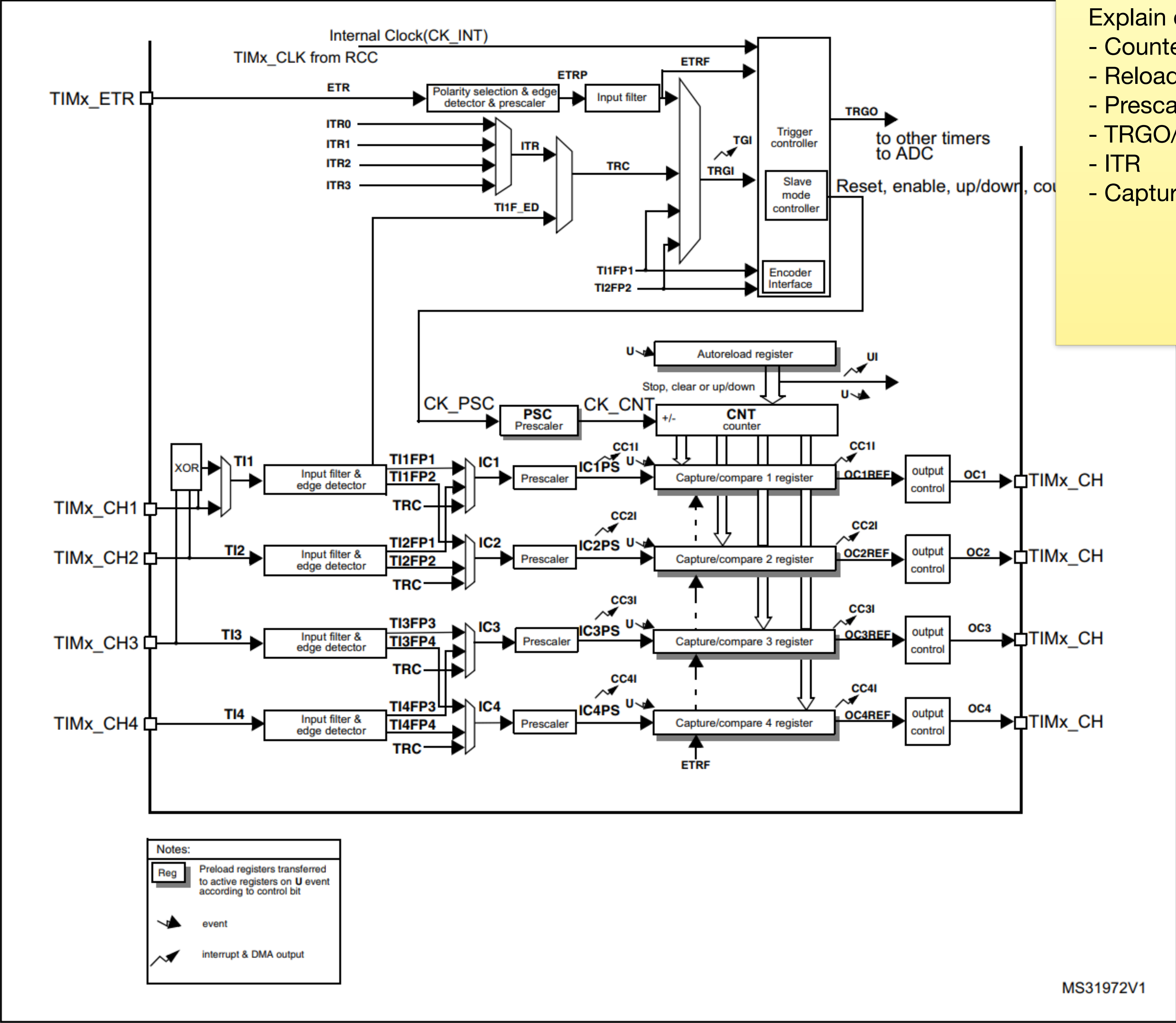
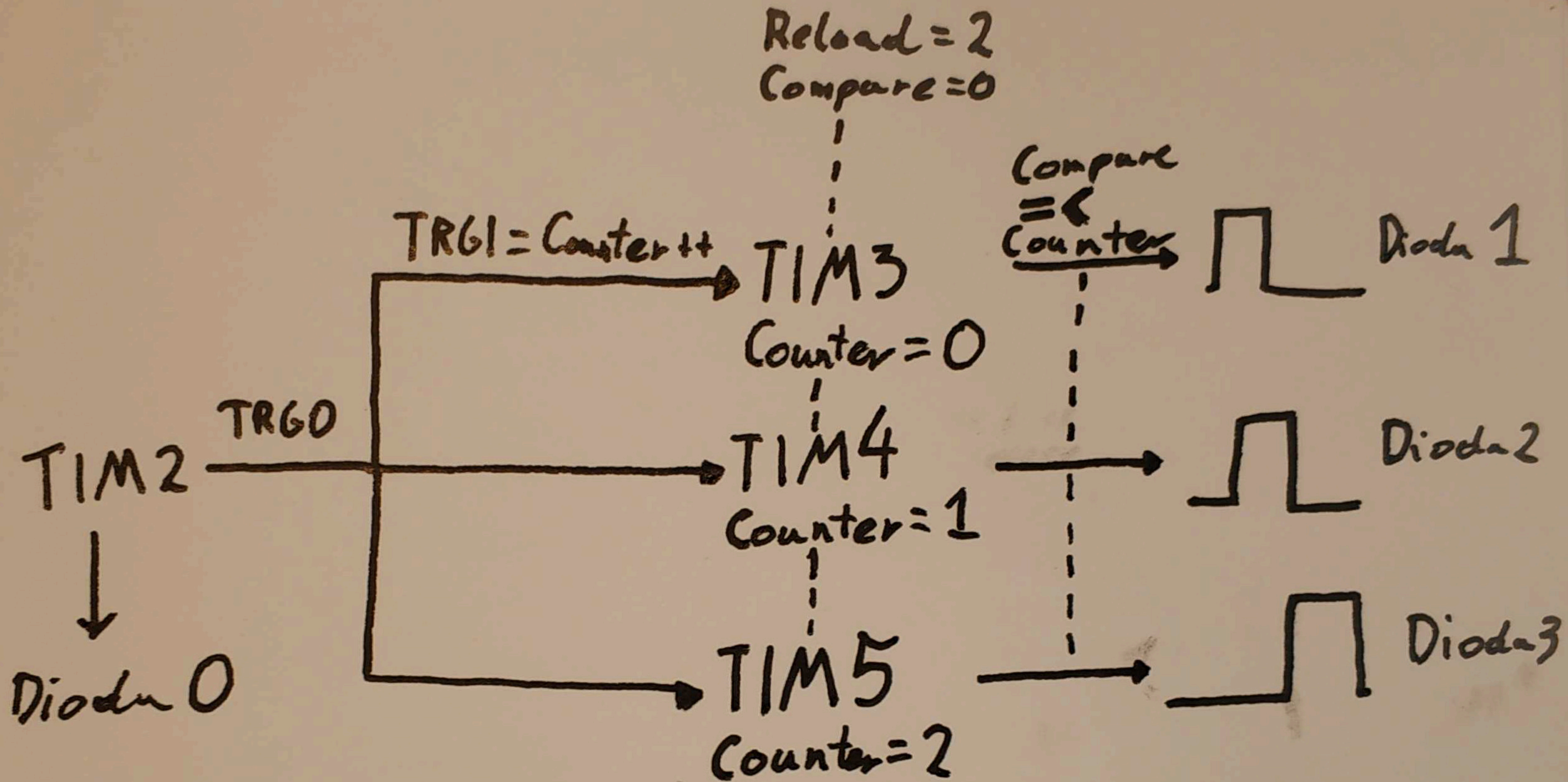


Figure 87. General-purpose timer block diagram



- Explain quickly:
- Counter
 - Reload
 - Prescaler
 - TRGO/TRGI
 - ITR
 - Capture Compare Register



13.4	TIM2 to TIM5 registers	353
13.4.1	TIMx control register 1 (TIMx_CR1)	353
13.4.2	TIMx control register 2 (TIMx_CR2)	355
13.4.3	TIMx slave mode control register (TIMx_SMCR)	356
13.4.4	TIMx DMA/Interrupt enable register (TIMx_DIER)	358
13.4.5	TIMx status register (TIMx_SR)	359
13.4.6	TIMx event generation register (TIMx_EGR)	361
13.4.7	TIMx capture/compare mode register 1 (TIMx_CCMR1)	362
13.4.8	TIMx capture/compare mode register 2 (TIMx_CCMR2)	365
13.4.9	TIMx capture/compare enable register (TIMx_CCER)	366
13.4.10	TIMx counter (TIMx_CNT)	368
13.4.11	TIMx prescaler (TIMx_PSC)	368
13.4.12	TIMx auto-reload register (TIMx_ARR)	368
13.4.13	TIMx capture/compare register 1 (TIMx_CCR1)	369
13.4.14	TIMx capture/compare register 2 (TIMx_CCR2)	369

13.4.1 TIMx control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 15:10 Reserved, must be kept at reset value.

Bits 9:8 **CKD**: Clock division

This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and sampling clock used by the digital filters (ETR, Tlx),

00: $t_{DTS} = t_{CK_INT}$

01: $t_{DTS} = 2 \times t_{CK_INT}$

10: $t_{DTS} = 4 \times t_{CK_INT}$

11: Reserved

Bit 7 **ARPE**: Auto-reload preload enable

0: TIMx_ARR register is not buffered

1: TIMx_ARR register is buffered

Bits 6:5 **CMS**: Center-aligned mode selection

00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR).

01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting down.

10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting up.

11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set both when the counter is counting up or down.

Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1)

Bit 4 **DIR**: Direction

0: Counter used as upcounter

1: Counter used as downcounter

Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.

Bit 3 **OPM**: One-pulse mode

0: Counter is not stopped at update event

1: Counter stops counting at the next update event (clearing the bit CEN)

More important things I marked with colours so I could connect with each other.

Pinouts and pin description

STM32F401xD STM32F401xE		
Table 7. Legend/abbreviations used in the pinout table		
Name	Abbreviation	
Pin name	Unless otherwise specified in brackets below the pin name, the pin function during and after reset is the same as the actual pin name	
Pin type	S	Definition
	I	
	I/O	
	FT	
I/O structure	B	Supply pin
	NRST	Input only pin
Notes	Input/output pin	
Alternate functions	5 V tolerant I/O	
Additional functions	Dedicated BOOT0 pin	
	Bidirectional reset pin with embedded weak pull-up resistor	
	Functions selected through GPIOx_AFR registers	
	Functions directly selected/enabled through peripheral registers	

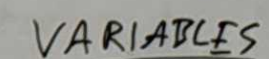
Table 8. STM32F401xD/xE pin definitions						
Pin Number	Pin name (function after reset) ⁽¹⁾	Pin type	I/O structure	Notes	Alternate functions	Additional functions
UQFN48	PE2	I/O	FT	-	SPI4_SCK, TRACECLK, EVENTOUT	-
WLCSP49	PE3	I/O	FT	-	TRACED0, EVENTOUT	-
LQFP100	PE4	I/O	FT	-	SPI4_NSS, TRACED1, EVENTOUT	-
LQFP100	PE5	I/O	FT	-	SPI4_MISO, TIM9_CH1, TRACED2, EVENTOUT	-
LQFP100	PE6	I/O	FT	-	SPI4_MOSI, TIM9_CH2, TRACED3, EVENTOUT	-
LQFP100	VSS	S	-	-	-	-
LQFP100	VDD	S	-	-	-	-
LQFP100	VBAT	S	-	-	-	-
LQFP100	PC13	I/O	FT (2) (3)	EVENTOUT,	-	RTC_TAMP1, RTC_OUT, RTC_TS



- Output open-drain with pull-up or pull-down capability
- Alternate function push-pull with pull-up or pull-down capability
- Alternate function push-pull with pull-up or pull-down capability
- Alternate function open-drain with pull-up or pull-down capability



TIM 2 = PAO CH1

$$T(M3) = PBO \text{ CH}_3$$
$$IM_4 = PB7 CH2$$
$$T(1M)5 = PA1 CH2$$


APB1 CLK TIMER = 84 MHz

TIM2 PRESCALER = 8400 - 1

TIM2 FREQ
↓
= 10 kHz

Reland

$-10k$
↓

1 ~ Output Freq

Composite Value = 5k

$$\sqrt{15}$$

NOT INCLUDED

Execution of plan

Code & Debugging




```

/**
 * @brief TIM2 Initialization Function
 * @param None
 * @retval None
 */
static void MX_TIM2_Init(void) {
    //START WITH THE COCK!?
    void *TIM2_ADDR = (0x40000000);
    uint16_t *TIM2_ENABLE = ((uint16_t *) (TIM2_ADDR + 0x00));
    uint16_t *TIM2_ENABLE_2 = ((uint16_t *) (TIM2_ADDR + 0x04));
    uint16_t *TIM2_COMPARE_MODE = ((uint16_t *) (TIM2_ADDR + 0x18));
    uint16_t *TIM2_COMPARE_OUTPUT_MODE = ((uint16_t *) (TIM2_ADDR + 0x20));
    uint32_t *TIM2_COUNTER_VALUE = ((uint32_t *) (TIM2_ADDR + 0x24));
    uint16_t *TIM2_PRESCALER_VALUE = ((uint16_t *) (TIM2_ADDR + 0x28));
    uint32_t *TIM2_RELOAD_VALUE = ((uint32_t *) (TIM2_ADDR + 0x2C));
    uint32_t *TIM2_COMPARE_VALUE = ((uint32_t *) (TIM2_ADDR + 0x34));

    __HAL_RCC_TIM2_CLK_ENABLE(); important

    *TIM2_ENABLE = 0b0000000010010001; // autoreload + counter enabled – at the end as I am scared
    *TIM2_ENABLE_2 = 0b0000000000100000; // Update = reload as external trigger //FOR LATER – change it to channel then

    *TIM2_PRESCALER_VALUE = 8400 - 1; //down to 10khz
    *TIM2_RELOAD_VALUE = 10000; //down to 1hz for PWM
    *TIM2_COUNTER_VALUE = *TIM2_RELOAD_VALUE; //as we have downcounting

    *TIM2_COMPARE_VALUE = 5000; //half of it
    *TIM2_COMPARE_MODE = 0b0000000001101000; // PWM1 MODE of compare + preload enabled (!?)
    *TIM2_COMPARE_OUTPUT_MODE = 0b0000000000000001; // ENABLE OUTPUT
}

```

```

/**
 * @brief TIM2 Initialization Function
 * @param None
 * @retval None
 */
static void MX_TIM2_Init(void) {
    __HAL_RCC_TIM2_CLK_ENABLE();
    TIMx_Init(
        timerAddr: (void *) 0x40000000,
        compareValueAddrOffset: 0x34,
        compareModeAddrOffset: 0x18,
        enable: TIMx_ENABLE, // autoreload + counter enabled
        enable2: 0b010u << 4u, // Using update as external trigger
        slaveMode: 0b0, // I'm master, not slave
        TIM2_PRESCALER, // PRESCALING AS A PART
        TIM2_RELOAD, // THIS VALUE WILL BE TAKEN FROM ADC SOON
        TIM2_RELOAD, // IT COULD BE EVEN 0, it doesn't matter
        compareValue: TIM2_RELOAD / 2, // half on, half dead
        compareMode: 0b110u << 4u | 0b1u << 3u, // PWM1 MODE of compare + preload enabled (!?)
        outputMode: 0b1); // Enable output for ch1
}

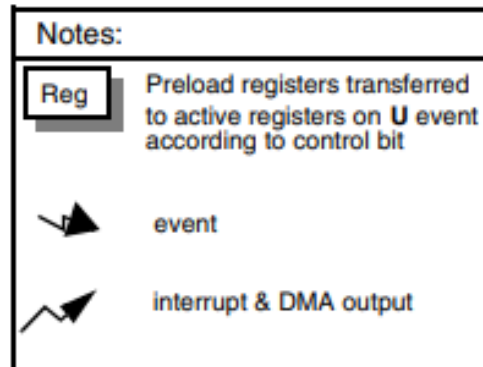
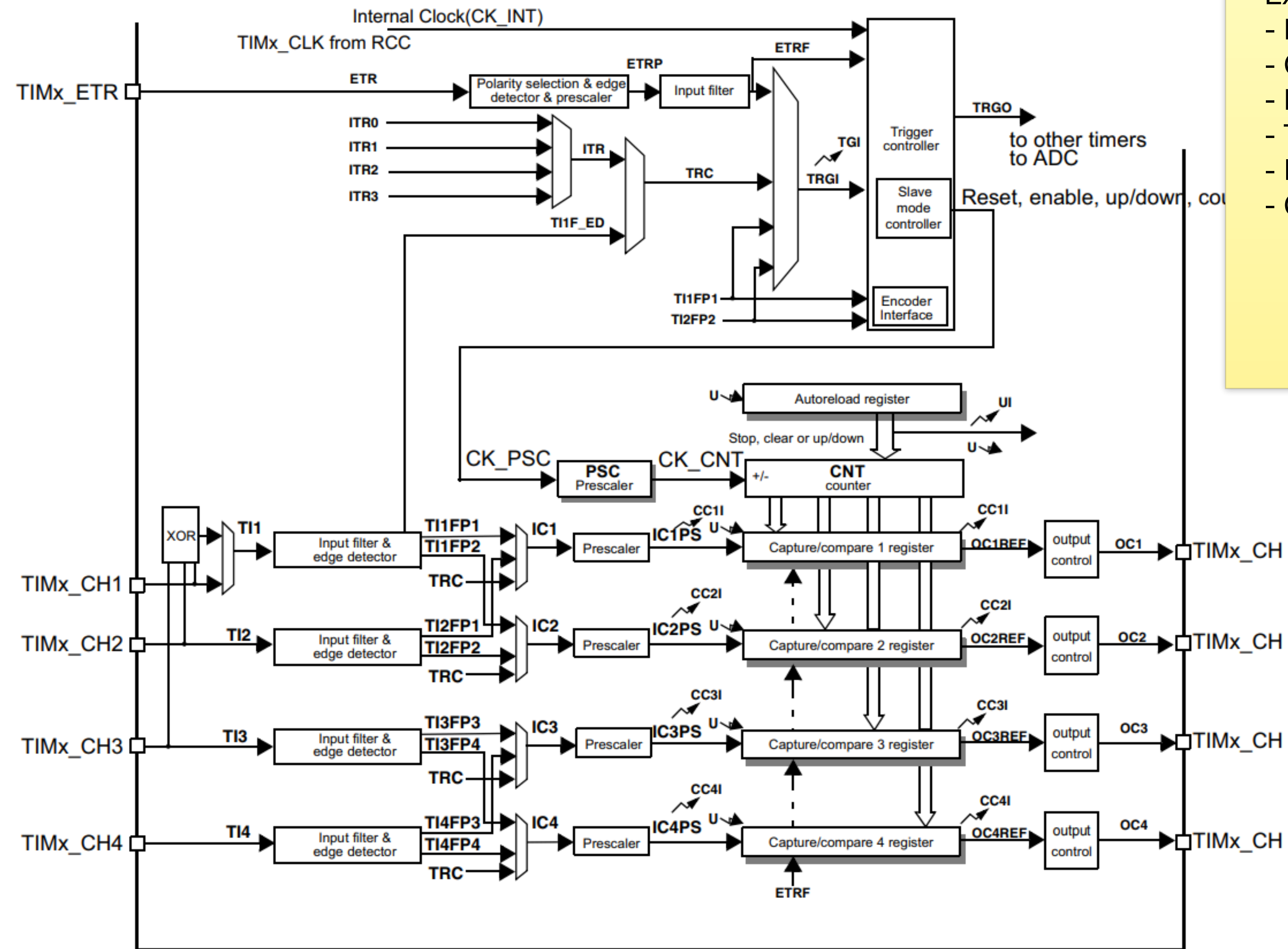
```


PROBLEM #1 - WHICH CHANNEL IS WHICH AF?

$T_{imx_Chx} - S \rightarrow GPIO_x$

REPETITION

Figure 87. General-purpose timer block diagram



Explain quickly:

- Prescaler
- Counter
- Reload
- TRGO/TRGI
- ITR
- Capture Compare Register

Documentation

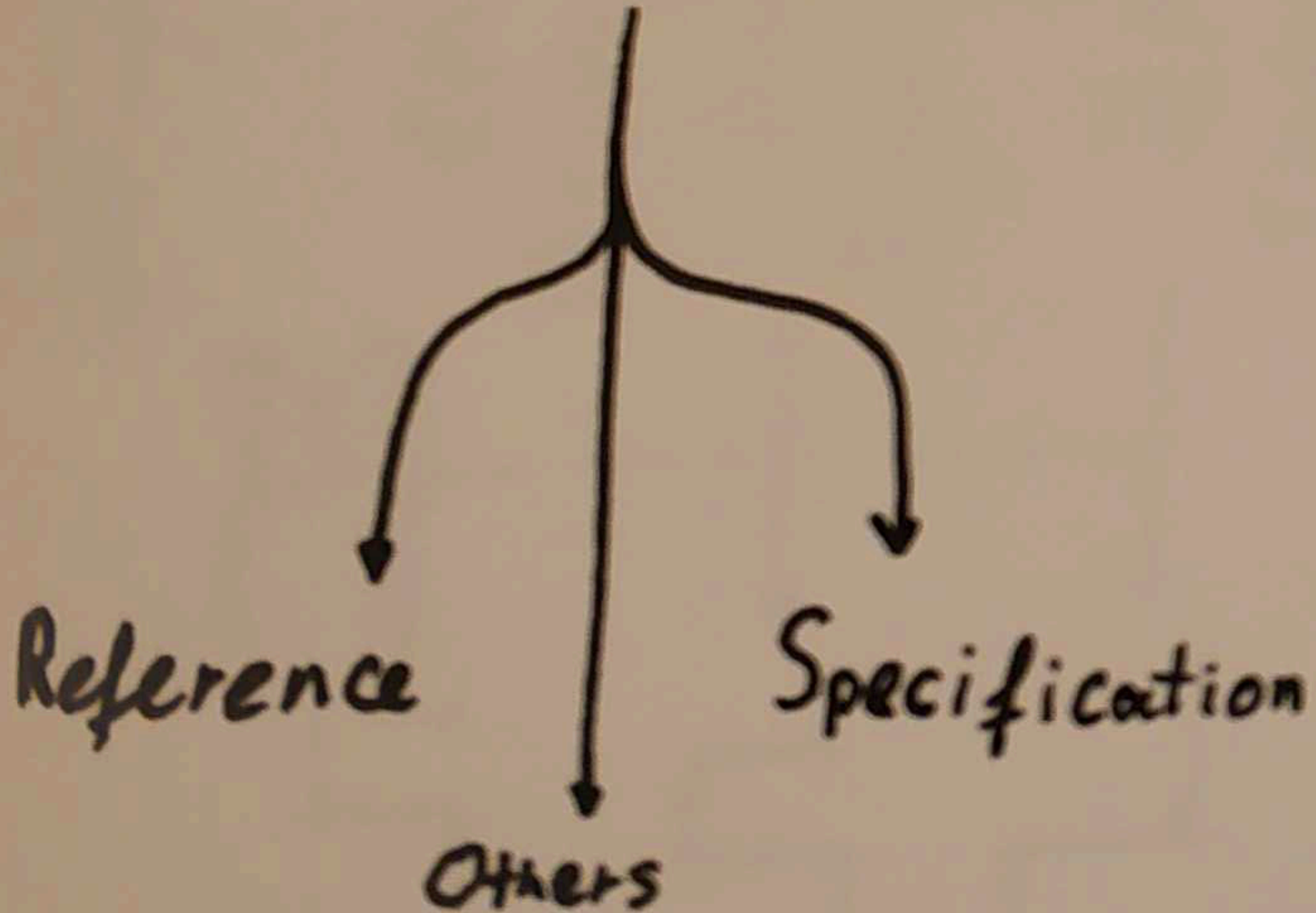
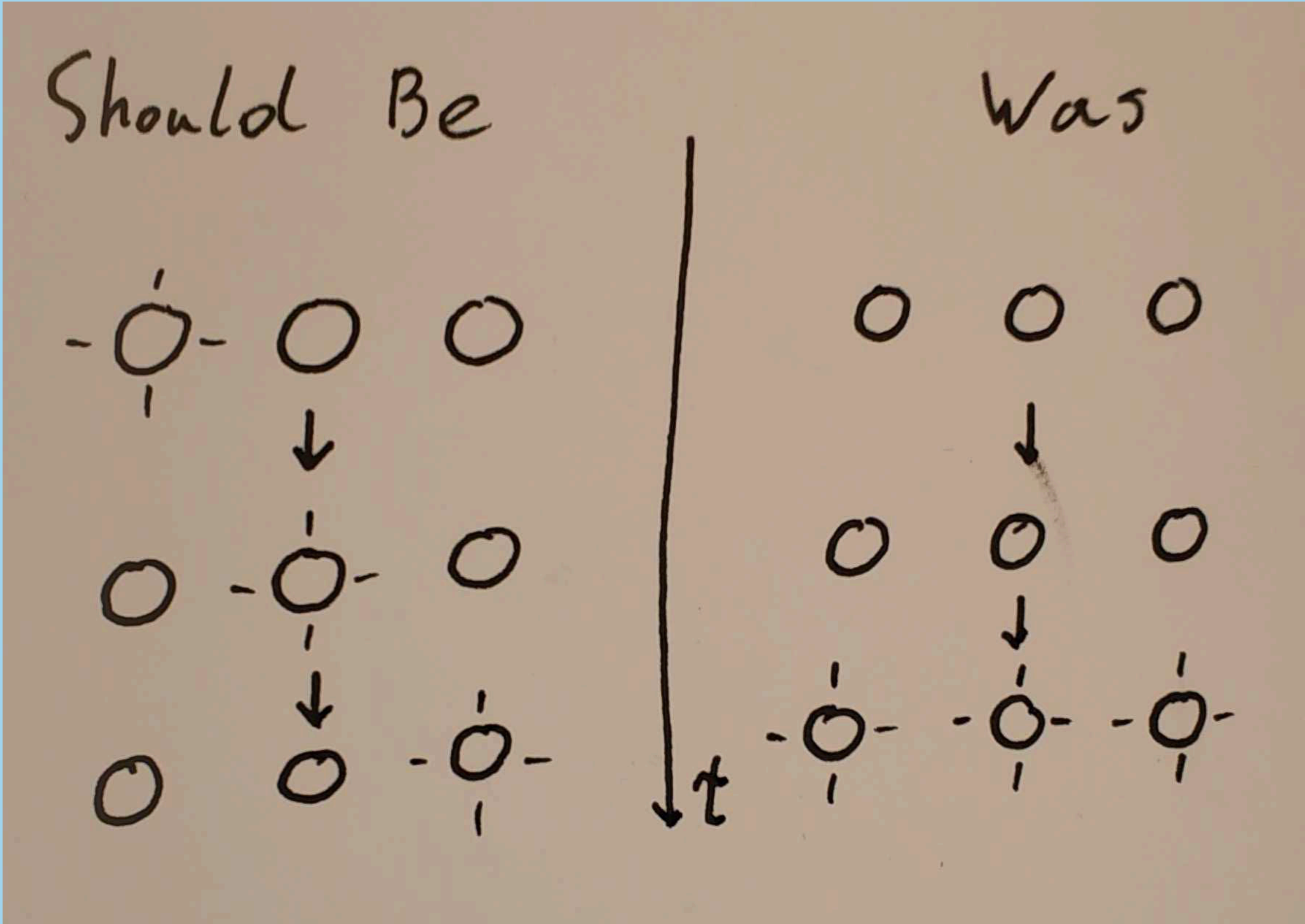


Table 8. STM32F401xD/xE pin definitions (continued)

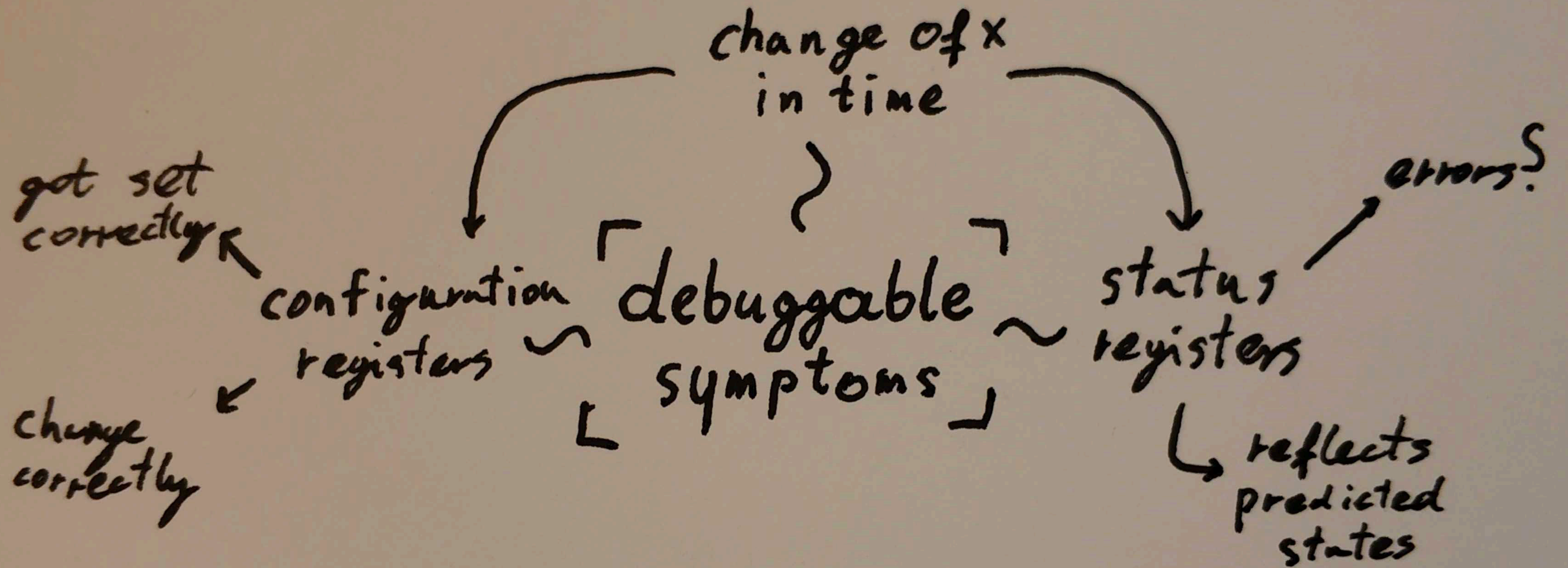
Pin Number					Pin name (function after reset) ⁽¹⁾	Pin type	I/O structure	Notes	Alternate functions	Additional functions
UQFN48	WLCSP49	LQFP64	LQFP100	UFBGA100						
13	E4	17	26	L3	PA3	I/O	FT	-	USART2_RX, TIM2_CH4, TIM5_CH4, TIM9_CH2, EVENTOUT	ADC1_IN3
-	-	18	27	-	VSS	S	-	-	-	-
-	-	19	28	-	VDD	S	-	-	-	-
-	-	-	-	E3	BYPASS_ REG	I	FT	-	-	-
14	G6	20	29	M3	PA4	I/O	FT	-	SPI1_NSS, SPI3_NSS/I2S3_WS, USART2_CK, EVENTOUT	ADC1_IN4
15	F5	21	30	K4	PA5	I/O	FT	-	SPI1_SCK, TIM2_CH1/TIM2_ETR, EVENTOUT	ADC1_IN5
16	F4	22	31	L4	PA6	I/O	FT	-	SPI1_MISO, TIM1_BKIN, TIM3_CH1, EVENTOUT	ADC1_IN6
17	F3	23	32	M4	PA7	I/O	FT	-	SPI1_MOSI, TIM1_CH1N, TIM3_CH2, EVENTOUT	ADC1_IN7
-	-	24	33	K5	PC4	I/O	FT	-	EVENTOUT	ADC1_IN14
-	-	25	34	L5	PC5	I/O	FT	-	EVENTOUT	ADC1_IN15
18	G5	26	35	M5	PB0	I/O	FT	-	TIM1_CH2N, TIM3_CH3, EVENTOUT	ADC1_IN8
19	G4	27	36	M6	PB1	I/O	FT	-	TIM1_CH3N, TIM3_CH4, EVENTOUT	ADC1_IN9

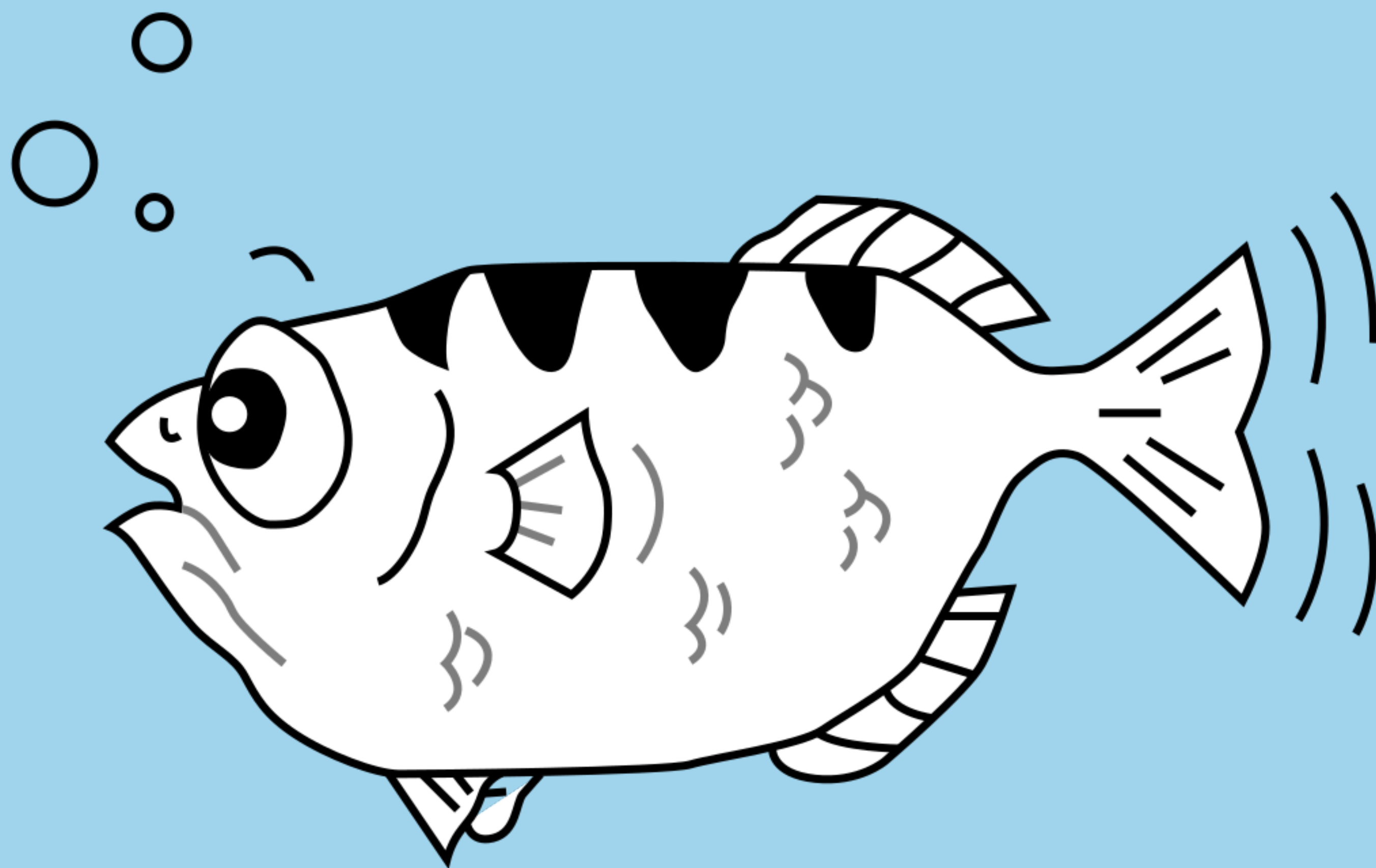
PROBLEM #2 - DIODES WERE DESYNCHRONISED



جی جی جی

PROBLEM






```
(gdb) print /t *((uint16_t*) 0x40000000)
$20 = 10001
(gdb) print /t *((uint16_t*) 0x40000004)
$21 = 100000
(gdb) print /t *((uint16_t*) 0x40000008)
$22 = 0
(gdb) print /t *((uint16_t*) 0x400000014)
warning: value truncated
$23 = 1100011011001
```

Tim2 configuration registers.

Checked configured registers.

```
*((uint32_t *) 0xE0042008) |= 0b1111u; // TIM2-5 STOP
```

0xE0042004	DBGMCU_CR	Reserved	0	DGB_CAN2_STOP
	Reset value		0	DBG_TIM7_STOP
			0	DBG_TIM6_STOP
			0	DBG_TIM5_STOP
			0	DBG_TIM8_STOP
			0	DBG_I2C2_SMBUS_TIMEOUT
			0	DBG_I2C1_SMBUS_TIMEOUT
			0	DBG_CAN1_STOP
			0	DBG_TIM4_STOP
			0	DBG_TIM3_STOP
			0	DBG_TIM2_STOP
			0	DBG_TIM1_STOP
			0	DBG_WWDG_STOP
			0	DBG_IWDGSTOP
			0	TRACE_MODE[1:0]
			0	
			0	TRACE_IOEN
		Reserved		
			0	DBG_STANDBY
			0	DBG_STOP
			0	DBG_SLEEP


```

134      //MX_USART2_UART_Init();
135      /* USER CODE BEGIN 2 */
136
137      /* USER CODE END 2 */
138
139      /* Infinite loop */
140      /* USER CODE BEGIN WHILE */
141      while (1) {
142          /* USER CODE END WHILE */
143
(gdb) l
144          /* USER CODE BEGIN 3 */
145      }
146      /* USER CODE END 3 */
147  }
148
149  /**
150   * @brief System Clock Configuration
151   * @retval None
152   */
153  void SystemClock_Config(void) {
(gdb) b 141
Breakpoint 1 at 0x80012b4: file /Users/psuwala/projects/STM32/PWMFROMSTM/Src/main.c, line 141.
(gdb) c
Continuing.
Note: automatically using hardware breakpoints for read-only addresses.

Breakpoint 1, main () at /Users/psuwala/projects/STM32/PWMFROMSTM/Src/main.c:141
141      while (1) {
(gdb) ^CQuit
(gdb) c
Continuing.

Breakpoint 1, main () at /Users/psuwala/projects/STM32/PWMFROMSTM/Src/main.c:141
141      while (1) {
(gdb) c
Continuing.

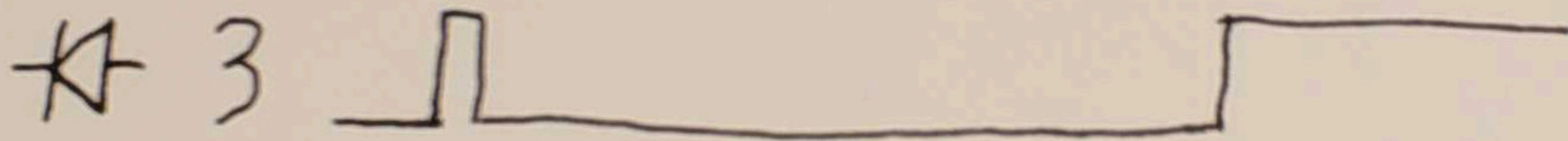
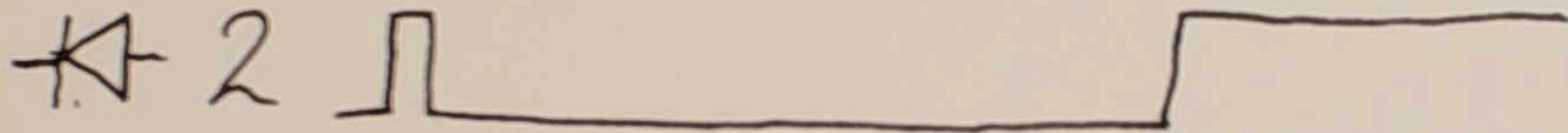
Breakpoint 1, main () at /Users/psuwala/projects/STM32/PWMFROMSTM/Src/main.c:141
141      while (1) {
(gdb) c
Continuing.

Breakpoint 1, main () at /Users/psuwala/projects/STM32/PWMFROMSTM/Src/main.c:141
141      while (1) {
(gdb) c
Continuing.

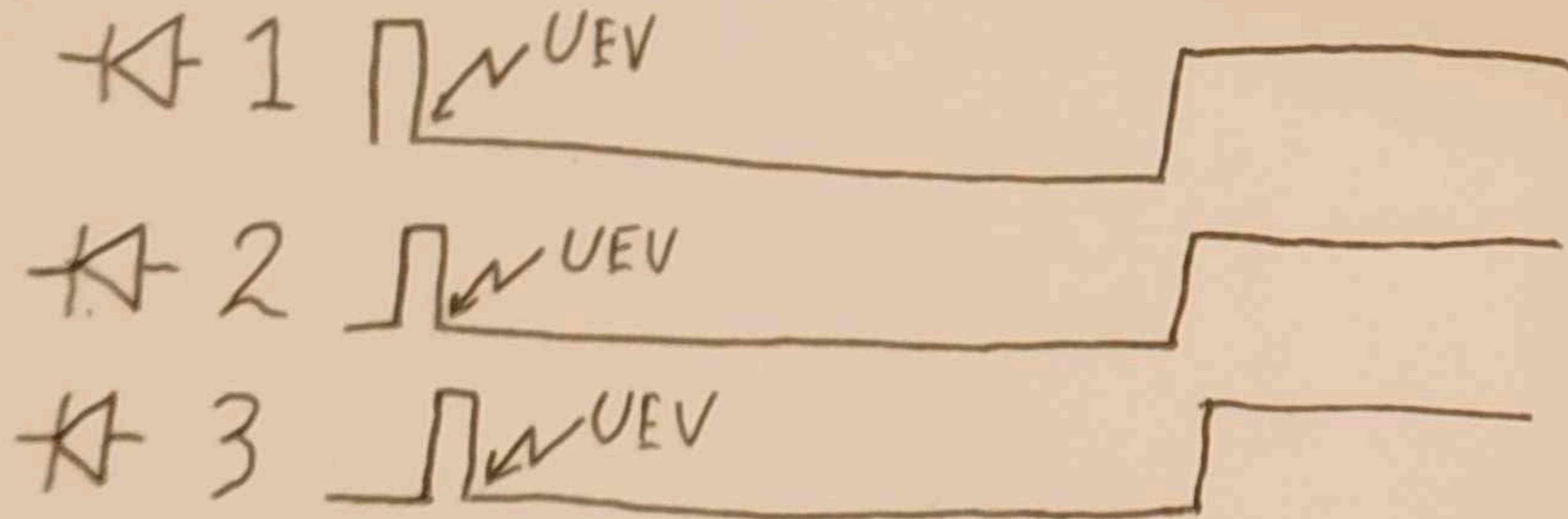
Breakpoint 1, main () at /Users/psuwala/projects/STM32/PWMFROMSTM/Src/main.c:141
141      while (1) {
(gdb) c
Continuing.

Breakpoint 1, main () at /Users/psuwala/projects/STM32/PWMFROMSTM/Src/main.c:141
141      while (1) {
(gdb) c
Continuing.

```



UEV = Update Event

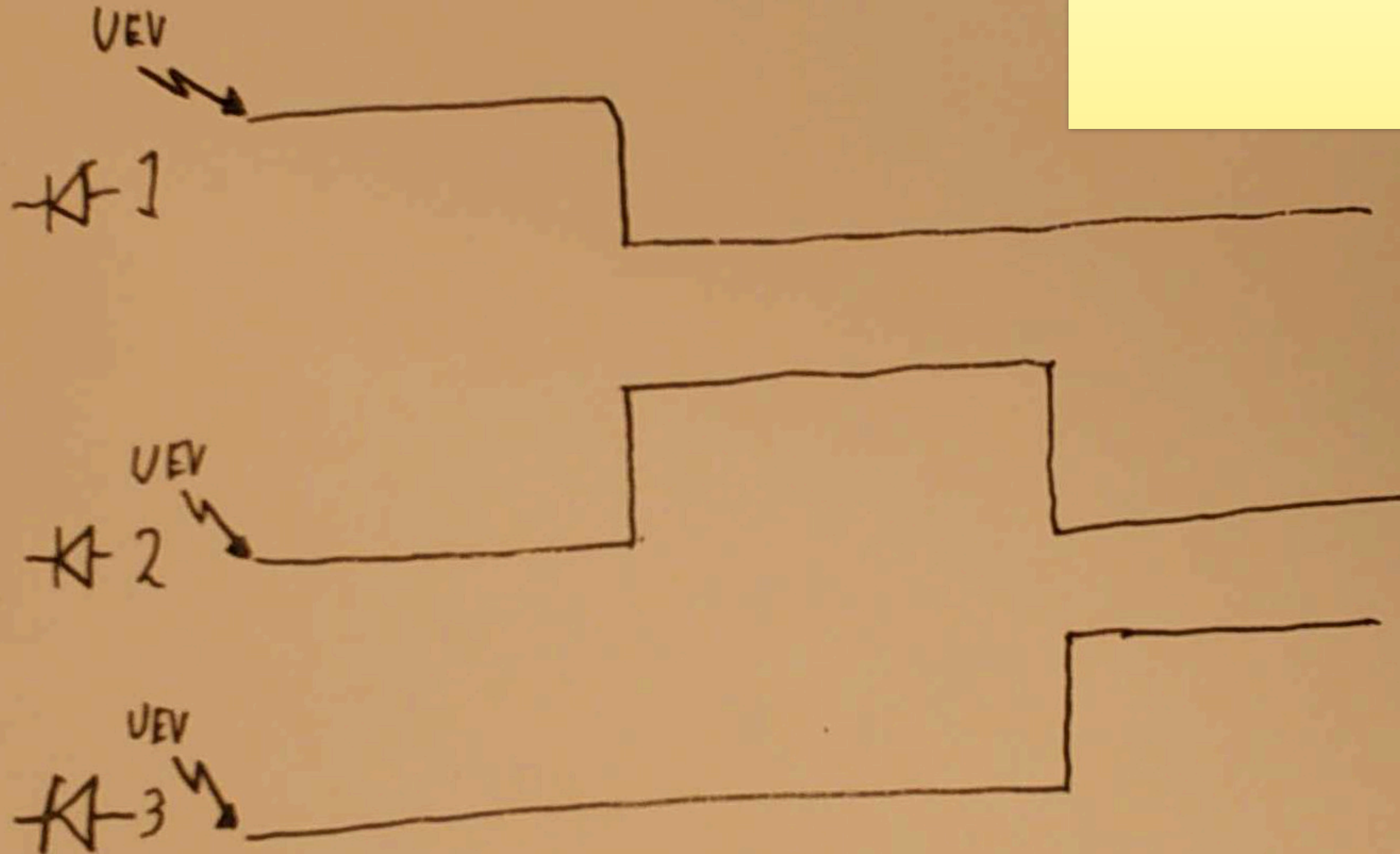


UEV = shadow registers → actual registers

shadow prescaler → actual prescaler

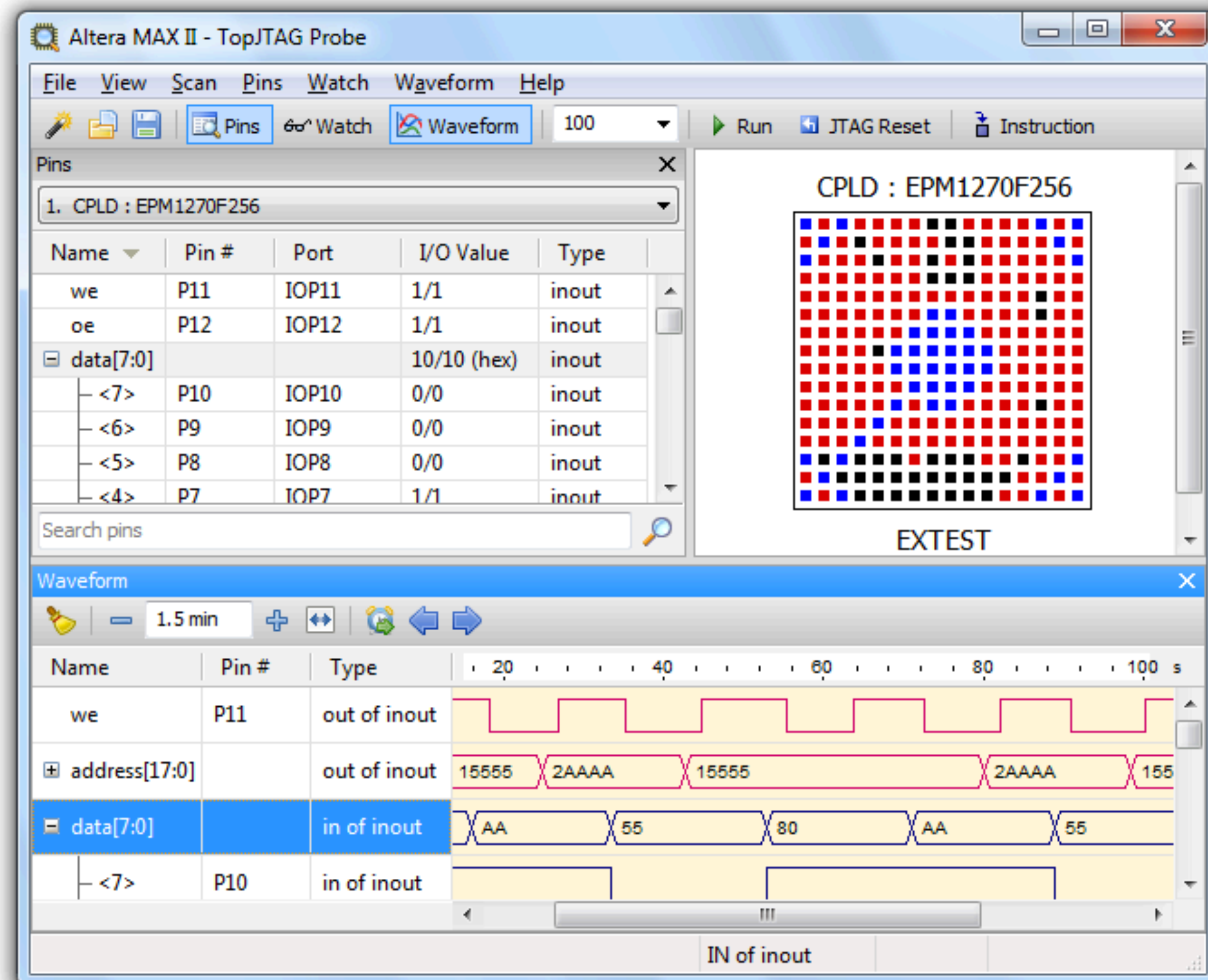
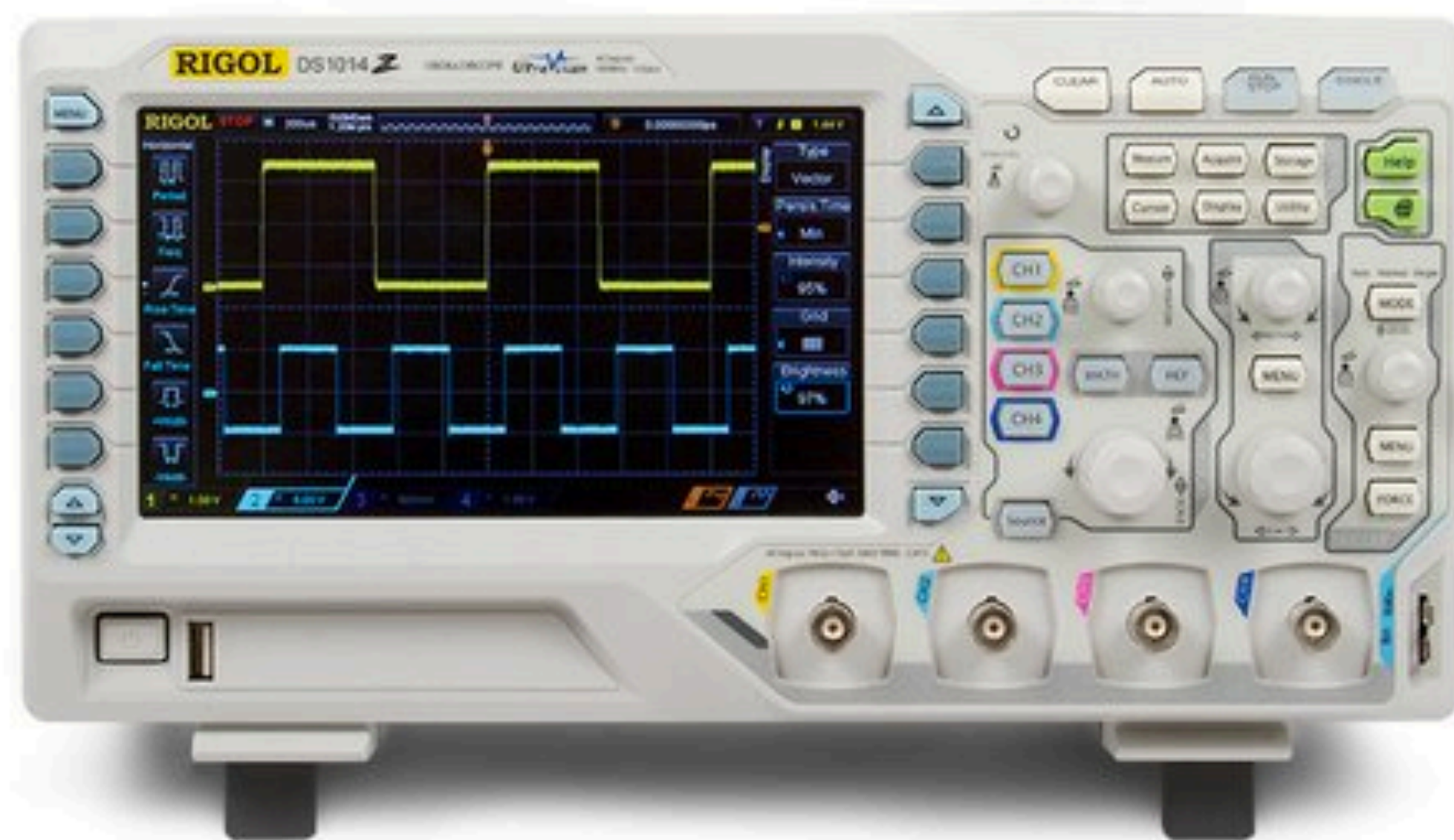
Self Induced UEV

I had to execute update before I started timers.



It was just updating one bit for every timer.

```
*TIM2_UPDATE |= 1u; //whatever, lets just update
```



ADC + DMA

Configuration and problems



Analog Digital Converter



DIRECT MEMORY ACCESS

DMA OFF



MEMCPY

01
10

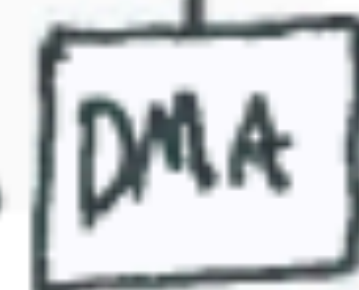
10
01

DMA ON



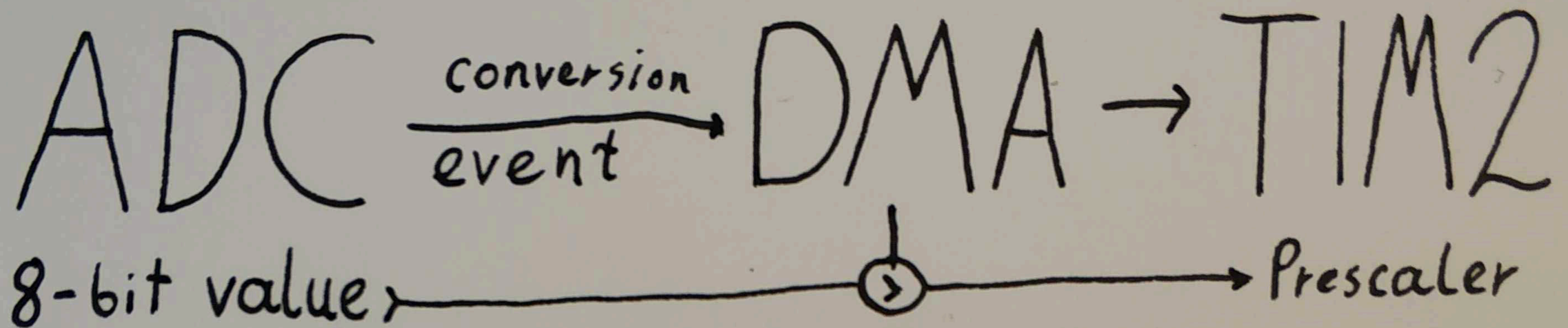
MAGIC

Flag



01
10

10
01



ADC $\xrightarrow[\text{event}]{\text{conversion}}$ DMA \rightarrow TIM2

8-bit value,

11111111

?



Prescaler

11111111

?

ADC $\xrightarrow[\text{event}]{\text{conversion}}$ DMA \rightarrow TIM2

8-bit value,

11111111

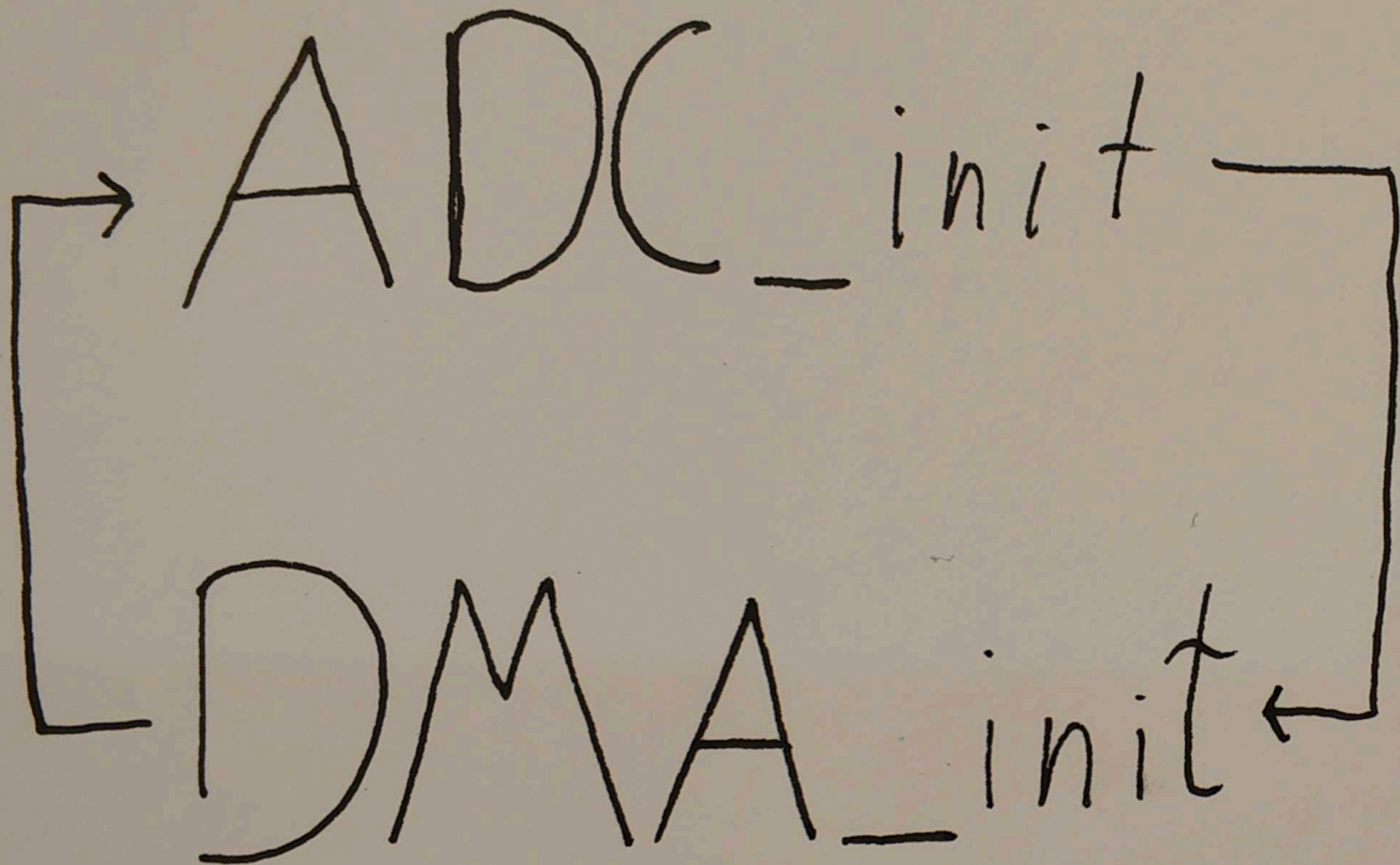
?

Status \rightarrow Overrun

Prescaler

11111111

?



Summary

What should we remember from this presentation

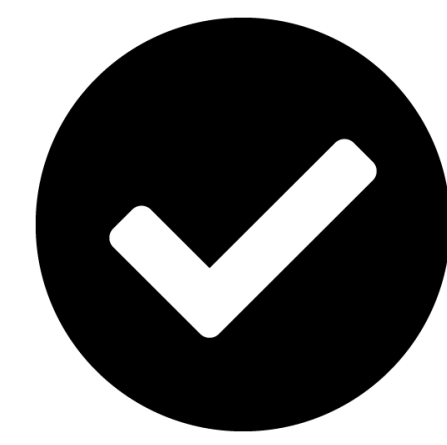




Tools



Common problems



Tips

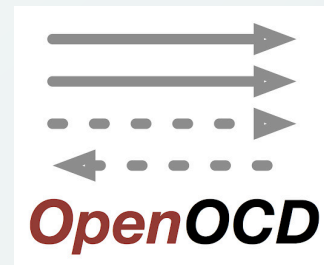


Tools



Clion or VSCode

Good IDE will help you develop code.



OpenOCD/GDB

Debugging will help you pinpoint the issues and go over obstacles.

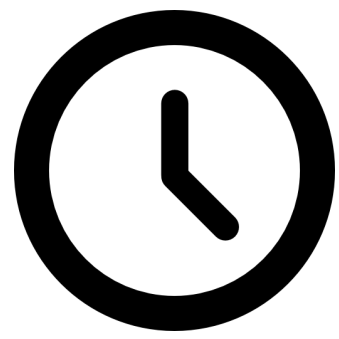


CubeMX

Helps to set up projects for stm products family.

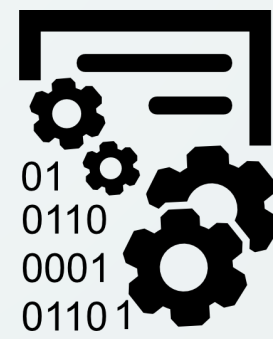


Common Problems



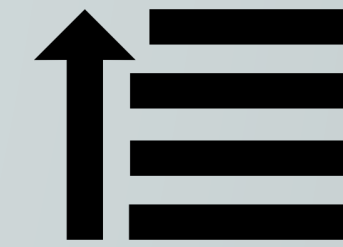
Set The Clock

All peripherals need their clock to be set before working on them.



Check Registers

Configuration and status registers may represent behaviour you didn't predict.



Order of Operations

Order of operations of registers may be crucial.

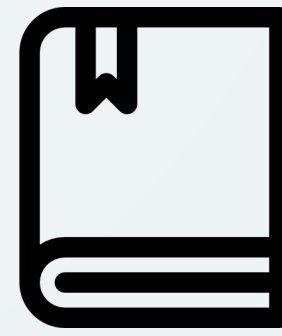


Tips



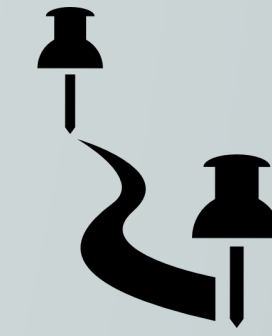
Organise Knowledge

Draw and organise your knowledge when learning.



Lurk More Documentation

Documentation has a lot of edge cases and behaviours saving days.



Plan Things

Understand what are you doing and why?





Thank you!