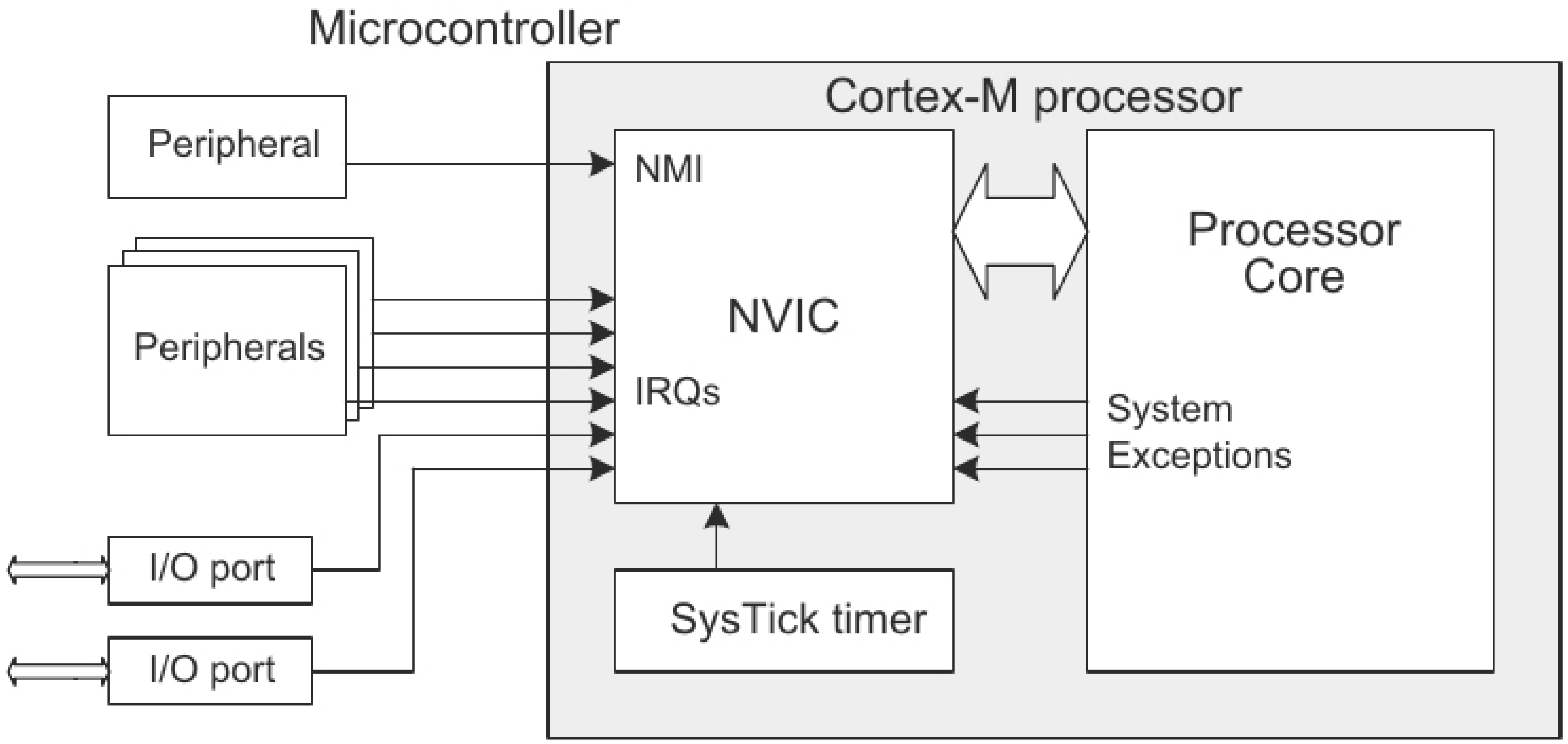
Microcomputer Engineering TMIK13 Lecture 6

TIMERS, GIT

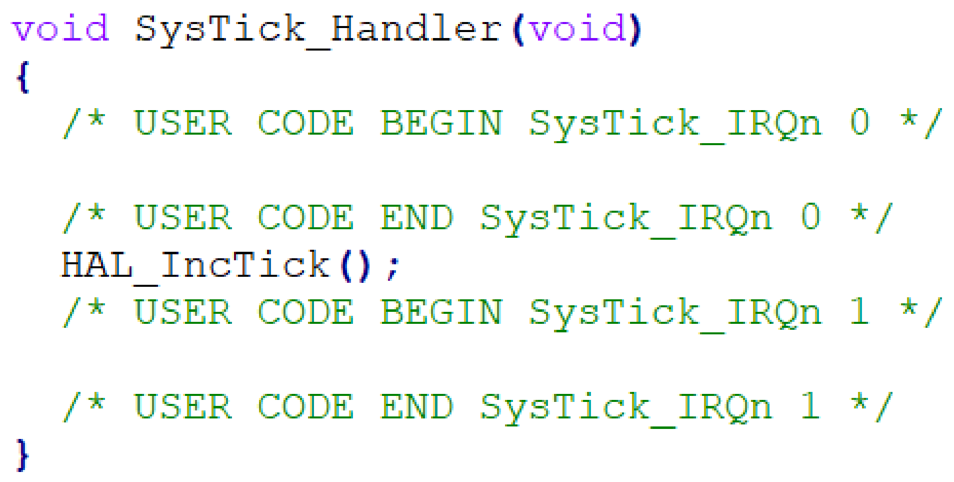
ANDREAS AXELSSON (ANDREAS.AXELSSON@JU.SE)

System Timer – SysTick



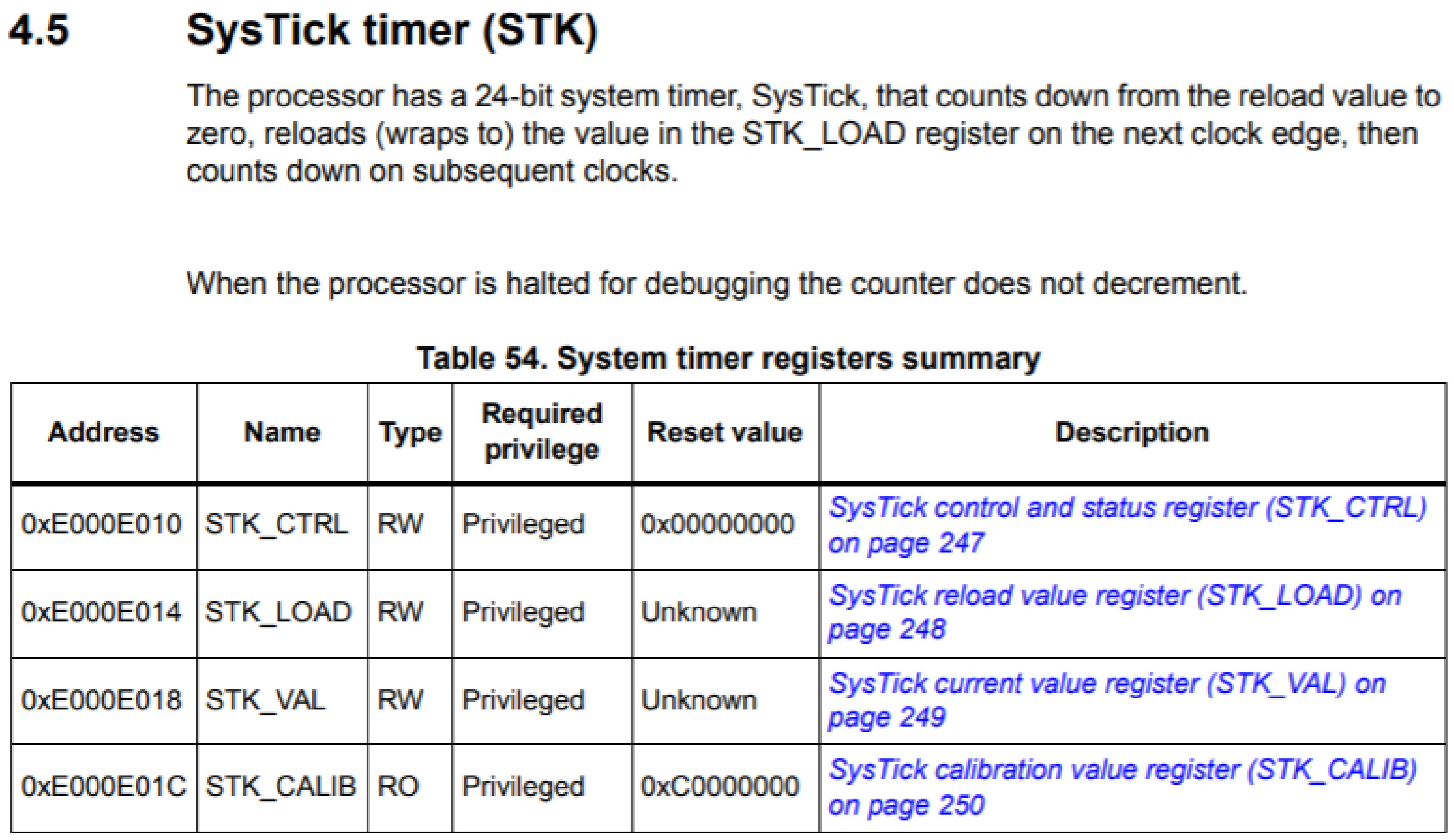
System Timer – SysTick

- System Timer is a standard timer inside the ARM Cortex-M
- It can be programmed to generate an interrupt each time it expires
- Often used to generate delays for system specific functions or task switching for operating systems.
- For instance, it is used to create a delay of a number of ticks with the HAL-function: `void HAL_Delay(uint32_t Delay)`



```
void SysTick_Handler(void)
{
  /* USER CODE BEGIN SysTick_IRQn 0 */

  /* USER CODE END SysTick_IRQn 0 */
  HAL_IncTick();
  /* USER CODE BEGIN SysTick_IRQn 1 */

  /* USER CODE END SysTick_IRQn 1 */
}
```

System Timer – SysTick



4.5 SysTick timer (STK)

The processor has a 24-bit system timer, SysTick, that counts down from the reload value to zero, reloads (wraps to) the value in the STK_LOAD register on the next clock edge, then counts down on subsequent clocks.

When the processor is halted for debugging the counter does not decrement.

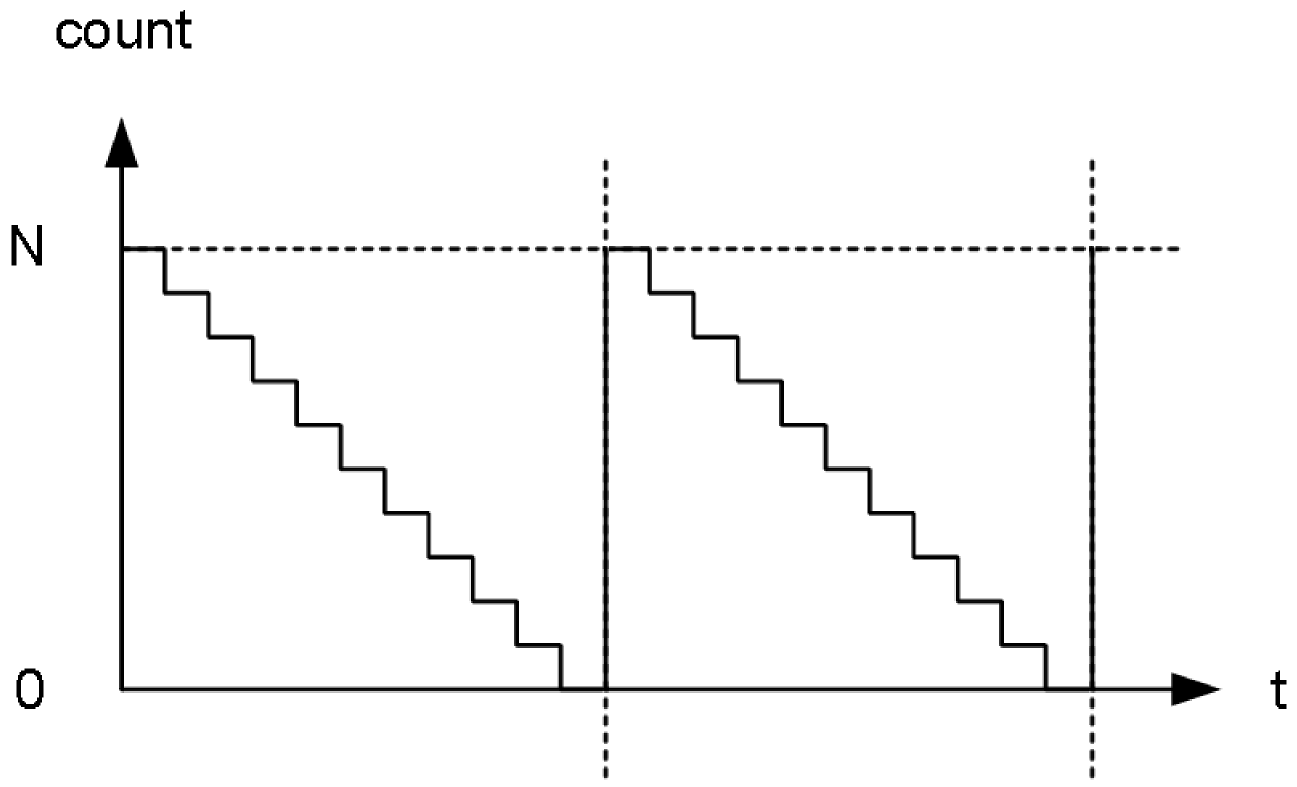
Table 54. System timer registers summary

Address	Name	Type	Required privilege	Reset value	Description
0xE000E010	STK_CTRL	RW	Privileged	0x00000000	*SysTick control and status register (STK_CTRL) on page 247*
0xE000E014	STK_LOAD	RW	Privileged	Unknown	*SysTick reload value register (STK_LOAD) on page 248*
0xE000E018	STK_VAL	RW	Privileged	Unknown	*SysTick current value register (STK_VAL) on page 249*
0xE000E01C	STK_CALIB	RO	Privileged	0xC0000000	*SysTick calibration value register (STK_CALIB) on page 250*

STM32 Cortex®-M4 MCUs and MPUs programming manual

System Timer – SysTick

$$\xrightarrow{f_{SysTick}} \boxed{\text{SysTick}} \xrightarrow{IRQ_{SysTick}}$$



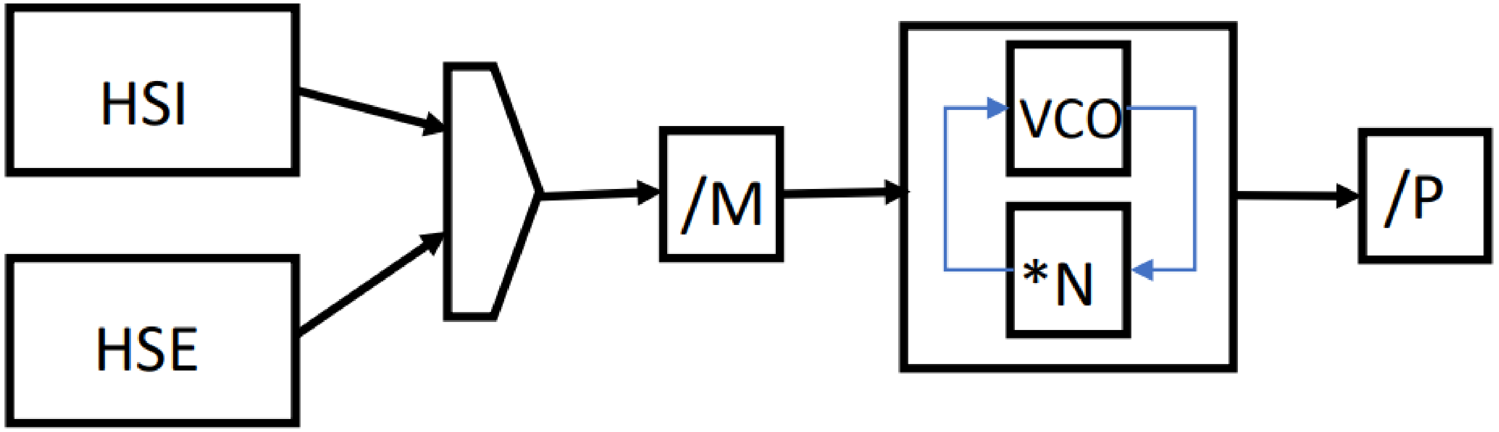
Clocks and important features

The clock sources can be both external and internal:

- HSI - High Speed Internal
- LSI - Low Speed Internal (32kHz)
- HSE - High Speed External (external crystal or oscillator)
- LSE - Low Speed External (32768 Hz external crystal or oscillator for RTC)

- PLL - Phased Locked Loop (Synthesize programmable frequencies)
- Prescaler - Divide a frequency with a programmable factor

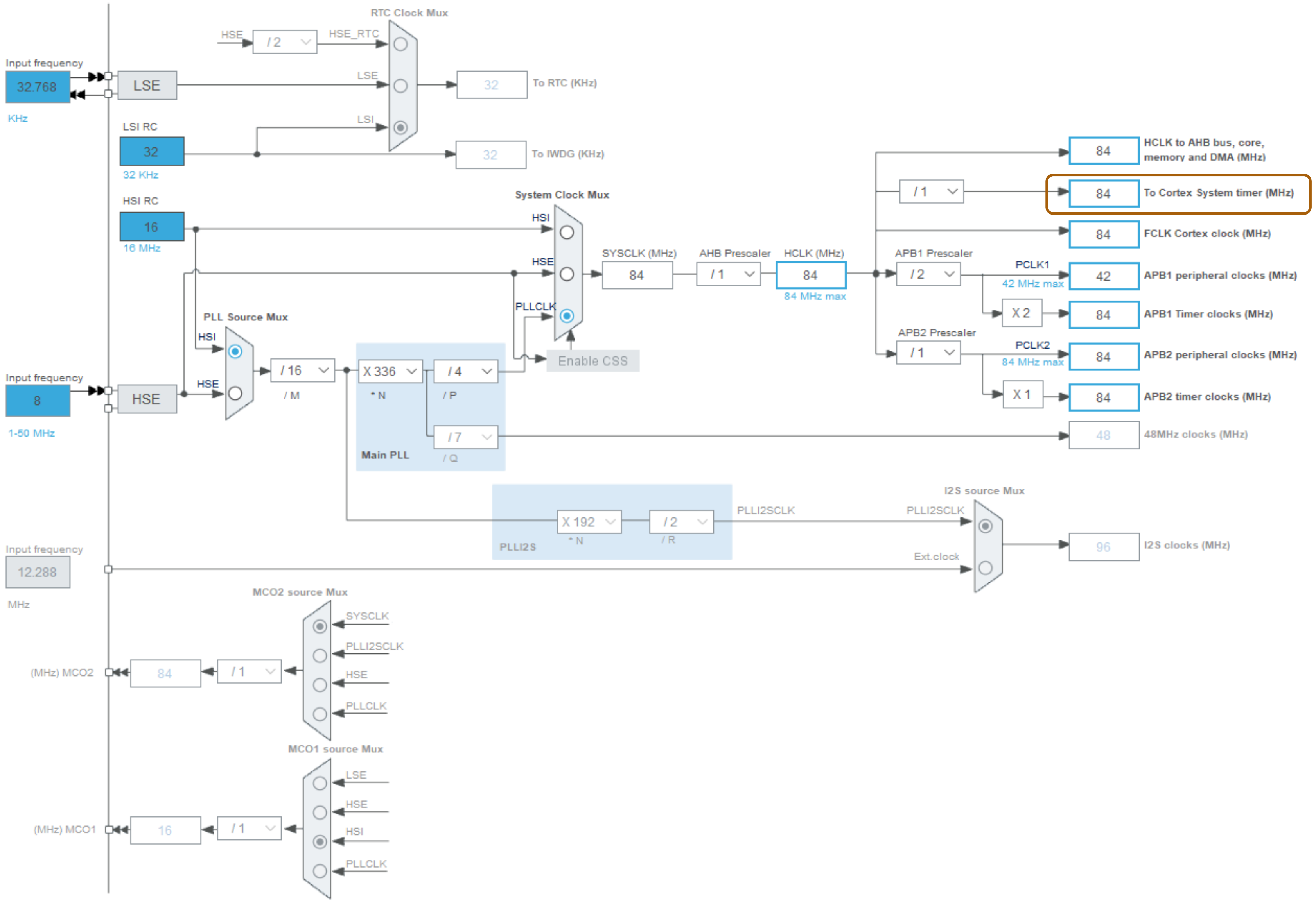
Phased Locked Loop – PLL



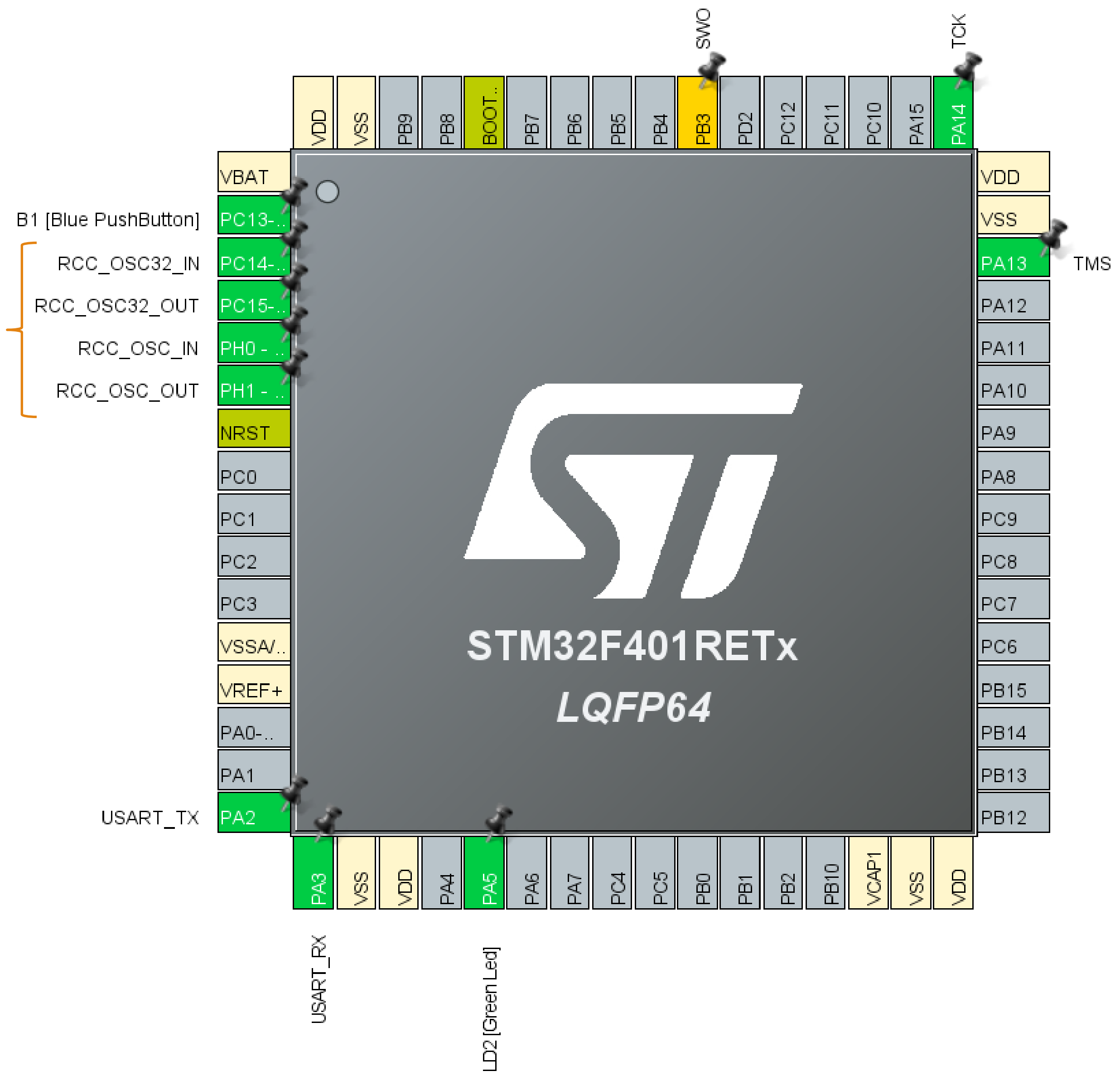
- $f_{(\text{VCO clock})} = f_{(\text{PLL clock input})} \times (\text{PLLN} / \text{PLLM})$
- $f_{(\text{PLL general clock output})} = f_{(\text{VCO clock})} / \text{PLLP}$

Use CubeMX to calculate M, N and P automatically

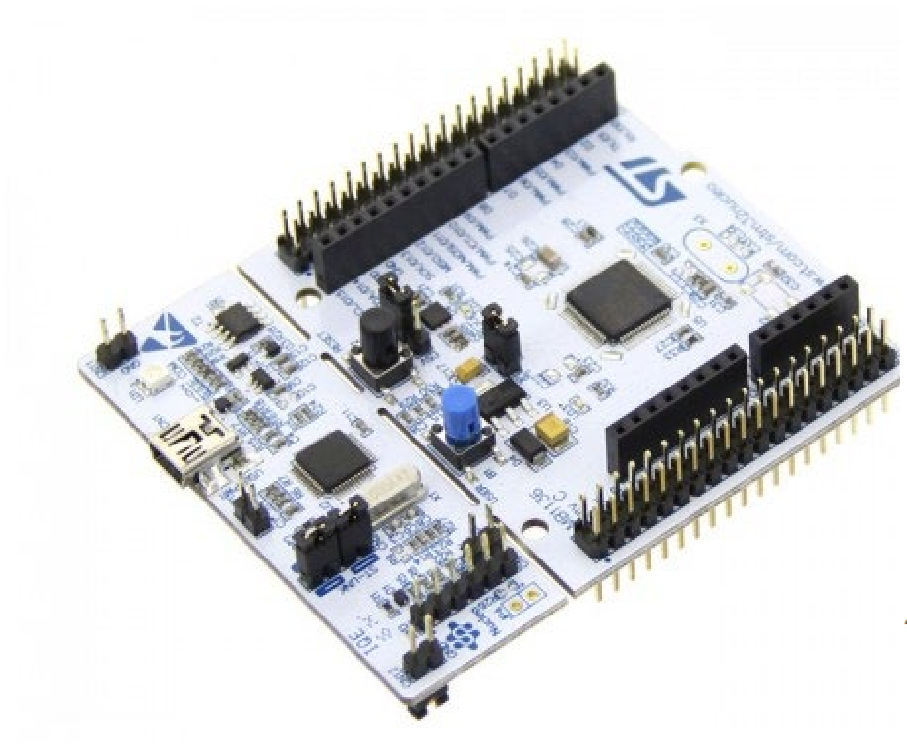
Clock Configuration

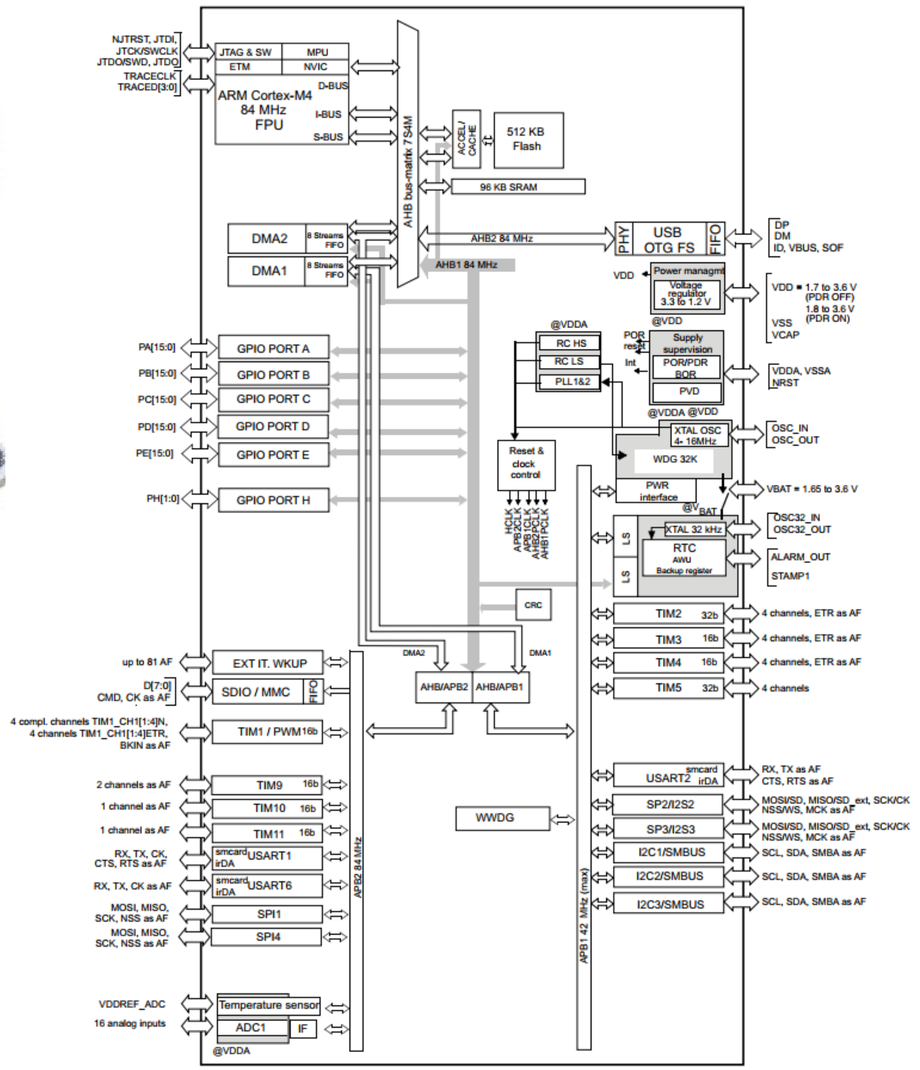


External Clock Inputs



Nucleo-64 STM32F401RE





Timers – STM32F401RE

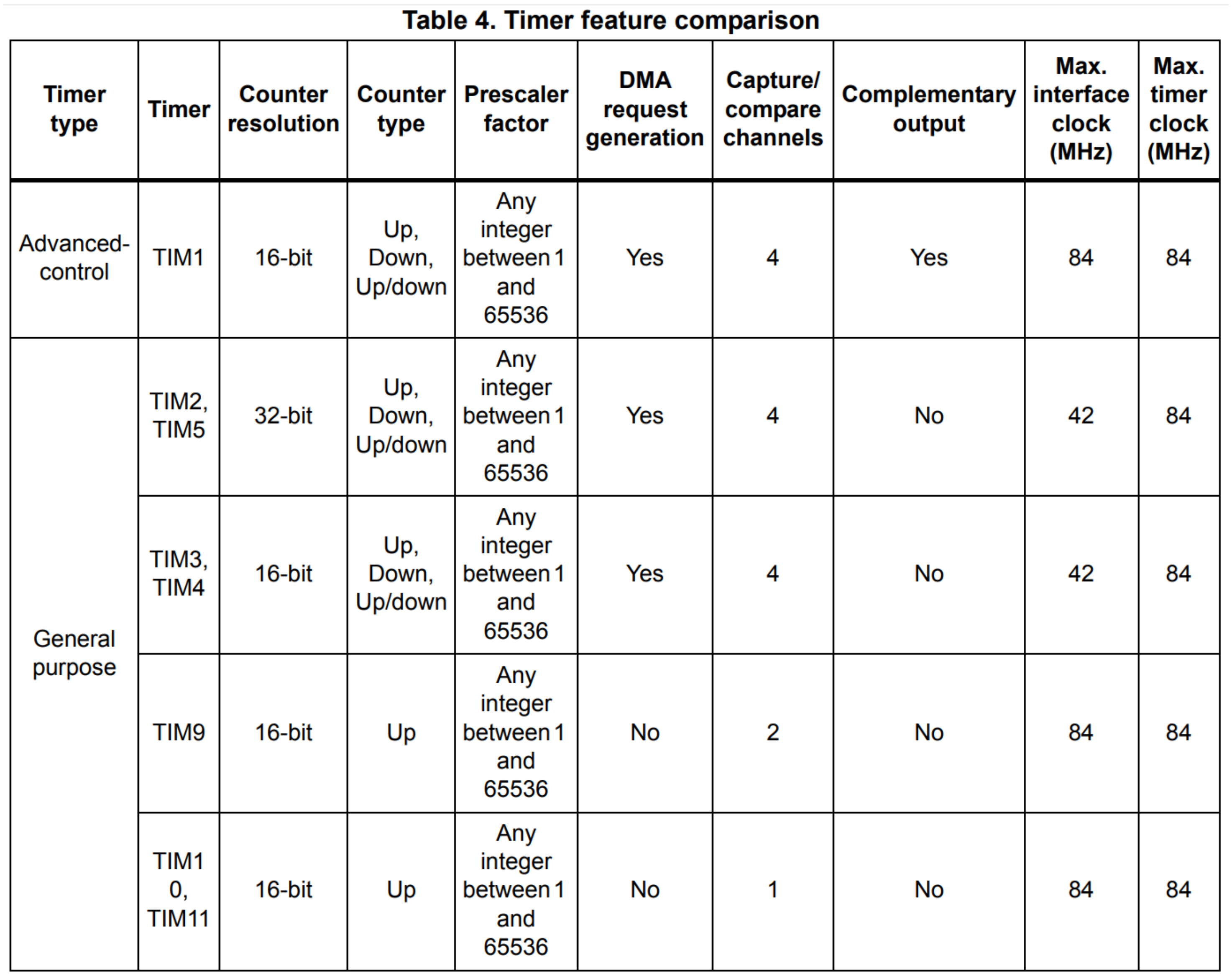
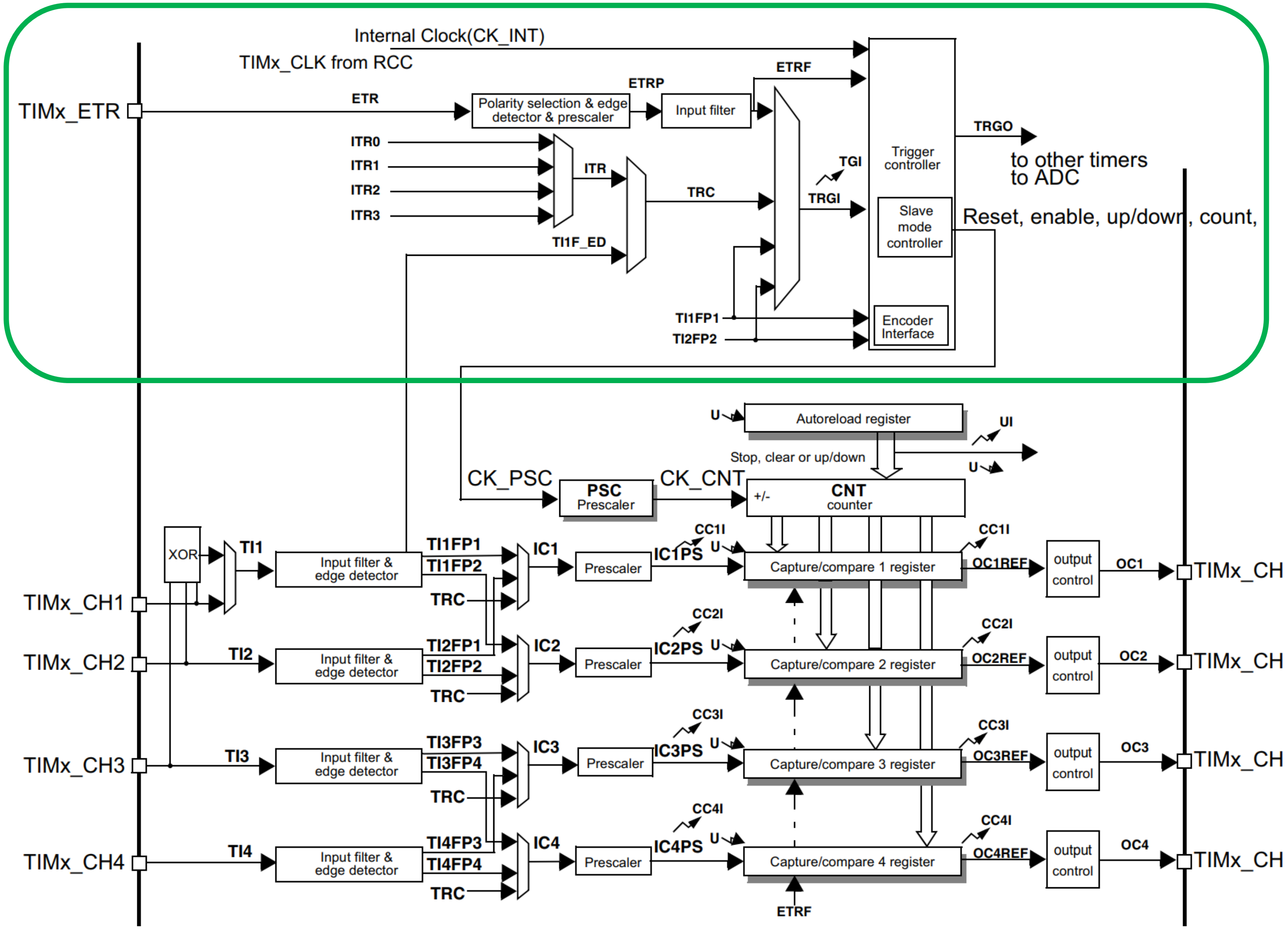


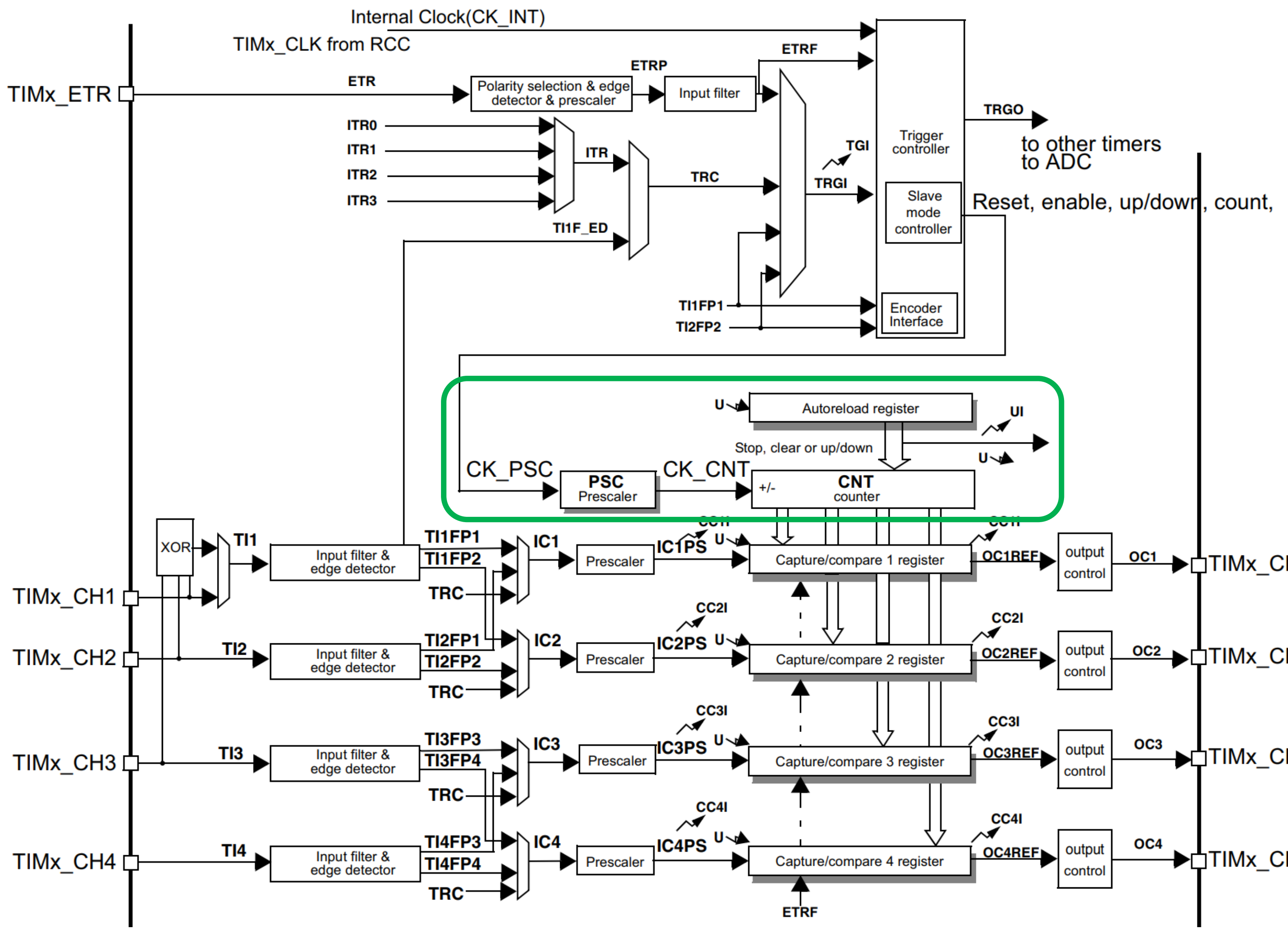
Table 4. Timer feature comparison

Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/ compare channels	Complementary output	Max. interface clock (MHz)	Max. timer clock (MHz)
Advanced-control	TIM1	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	Yes	84	84
General purpose	TIM2, TIM5	32-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM3, TIM4	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM9	16-bit	Up	Any integer between 1 and 65536	No	2	No	84	84
	TIM10, TIM11	16-bit	Up	Any integer between 1 and 65536	No	1	No	84	84

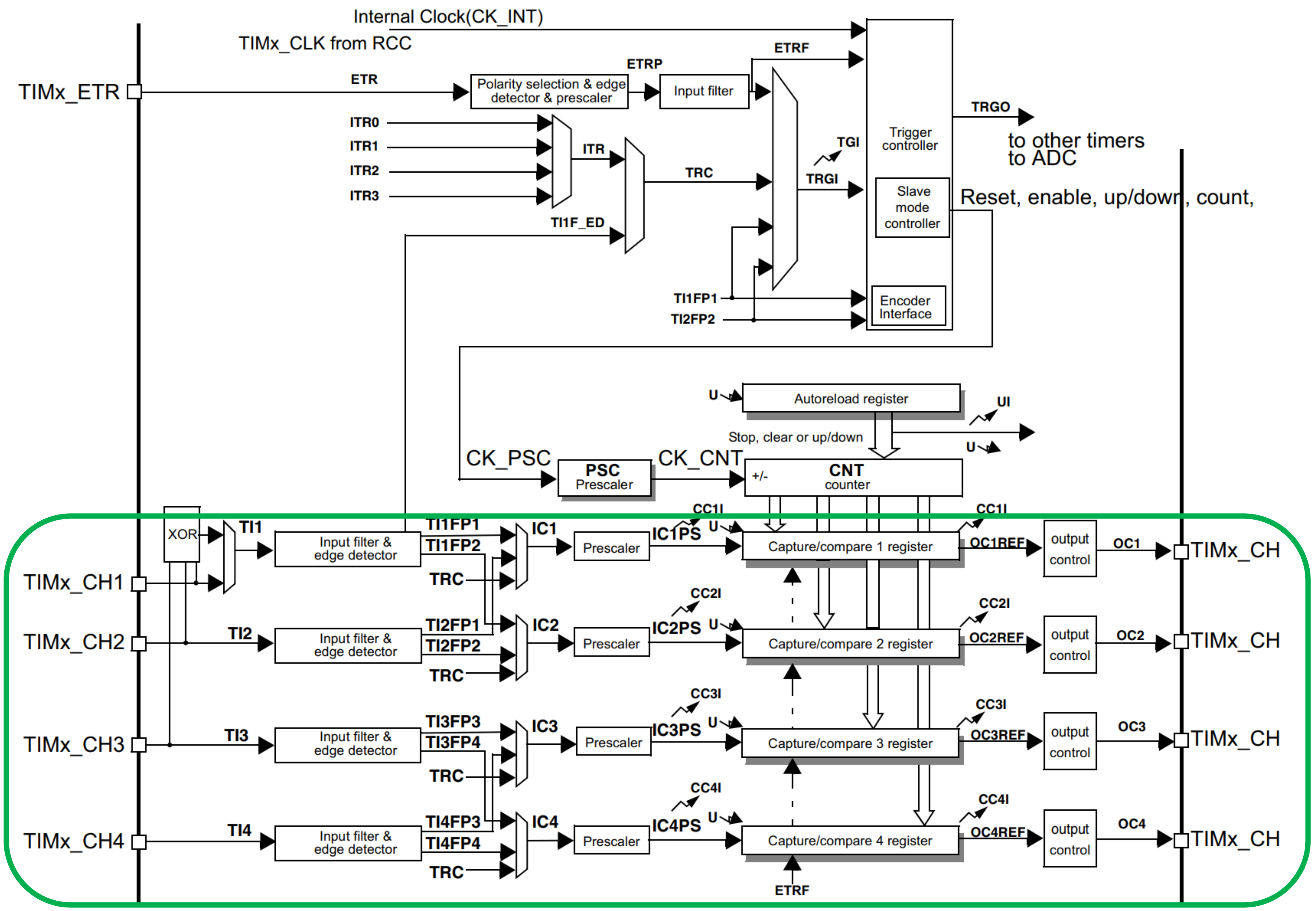
Timer – Block Schematics



Timer – Block Schematics



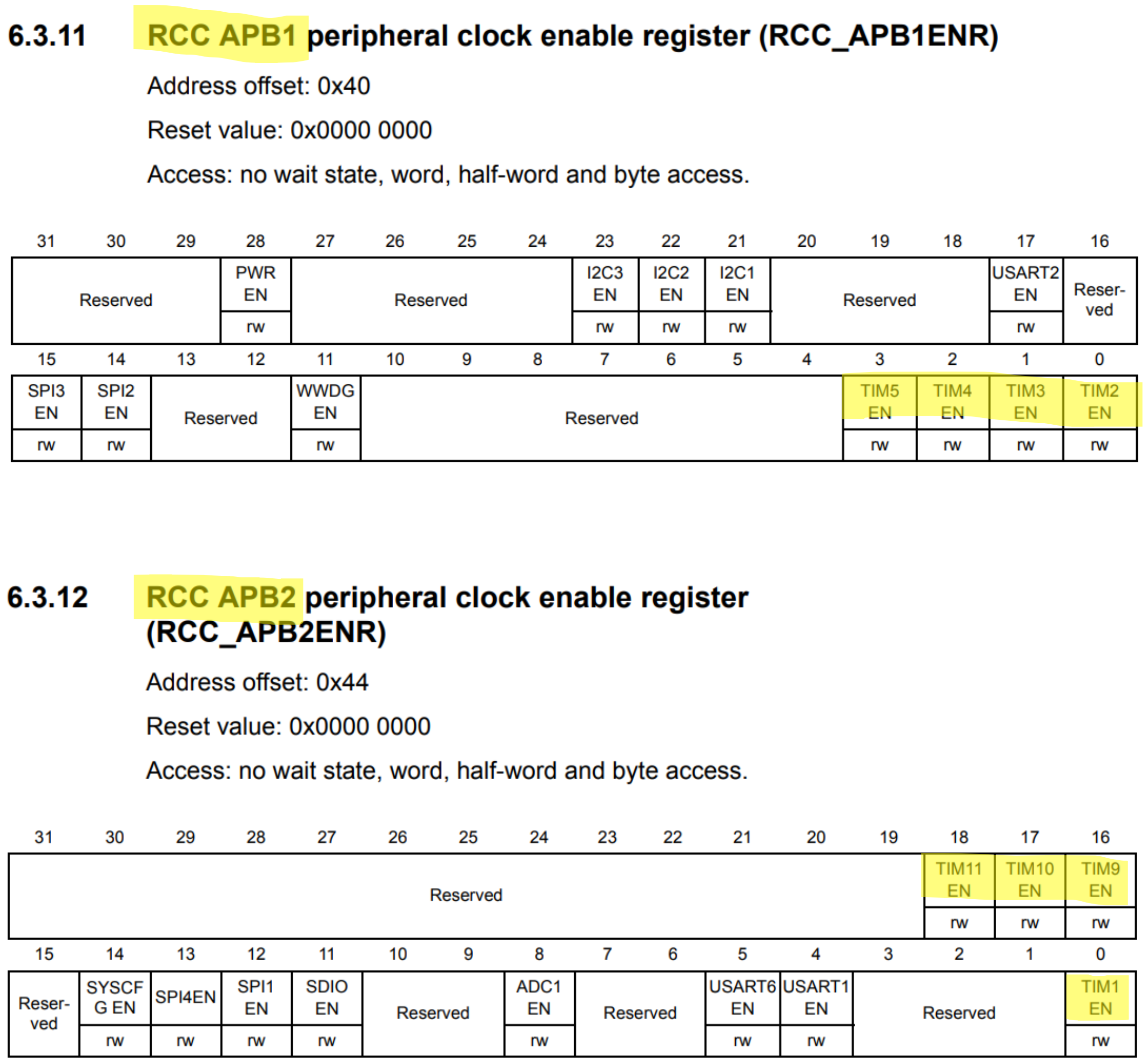
Timer – Block Schematics



Timers – Features

- Count
  - Up / Down or Up/Down
  - Internal Clock, external events, or encoder inputs
  - One Pulse or Continuous Reload
- Input Capture
  - Measure time / width of external events
- Output Compare
  - Generate Pulses
  - Pulse Width Modulation (PWM)
- Interrupts
- Trigger DMA

Timers – Clock Source



6.3.11 RCC APB1 peripheral clock enable register (RCC_APB1ENR)

Address offset: 0x40

Reset value: 0x0000 0000

Access: no wait state, word, half-word and byte access.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			PWR EN	Reserved				I2C3 EN	I2C2 EN	I2C1 EN	Reserved			USART2 EN	Reser-ved
			rw					rw	rw	rw				rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3 EN	SPI2 EN	Reserved		WWDG EN	Reserved							TIM5 EN	TIM4 EN	TIM3 EN	TIM2 EN
rw	rw			rw								rw	rw	rw	rw

6.3.12 RCC APB2 peripheral clock enable register (RCC_APB2ENR)

Address offset: 0x44

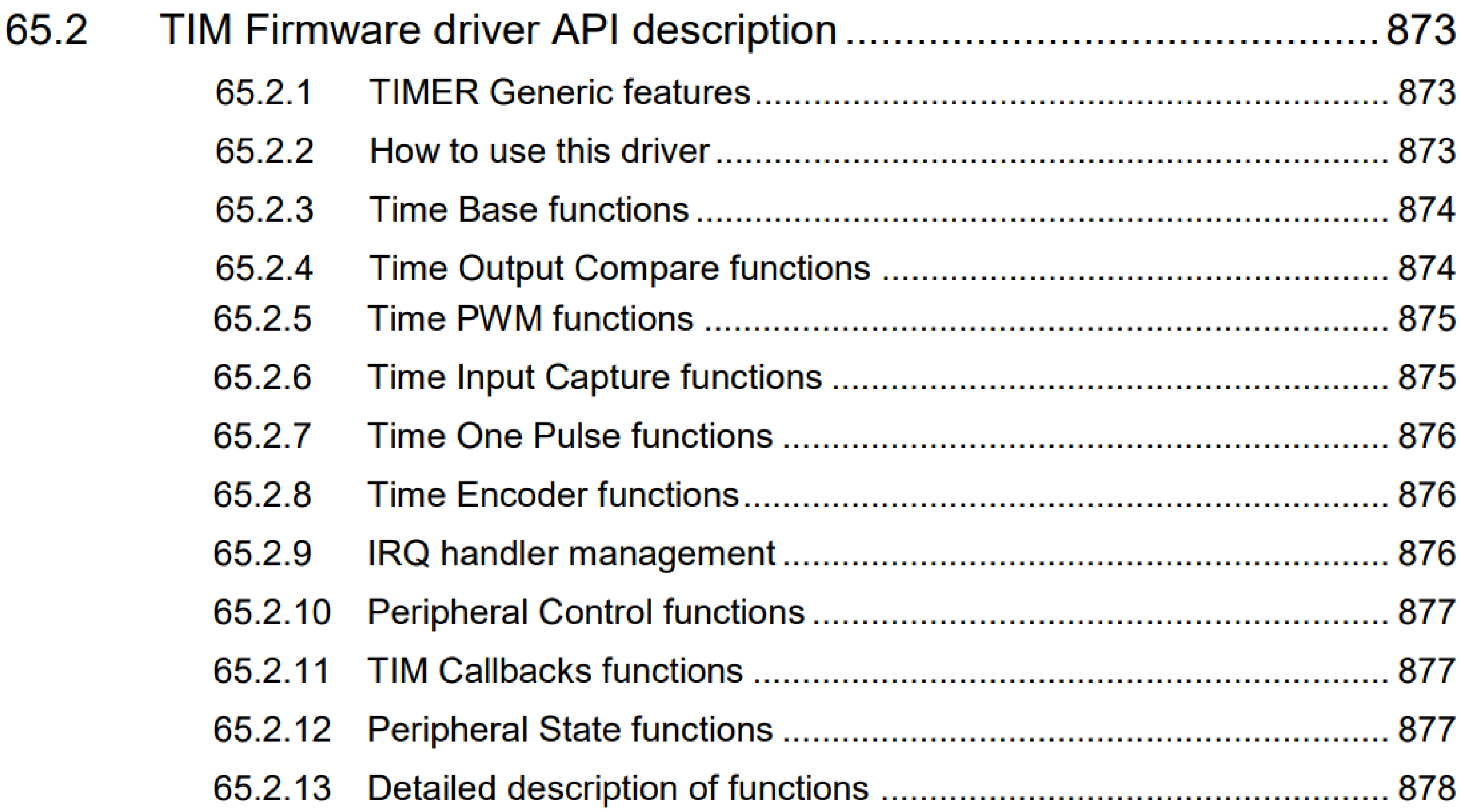
Reset value: 0x0000 0000

Access: no wait state, word, half-word and byte access.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													TIM11 EN	TIM10 EN	TIM9 EN
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reser-ved	SYSCFG EN	SPI4EN	SPI1 EN	SDIO EN	Reserved		ADC1 EN	Reserved		USART6 EN	USART1 EN	Reserved			TIM1 EN
	rw	rw	rw	rw			rw			rw	rw				rw

Timers – HAL Driver



Version Control – Git

Git is a tool developed 2005 by Linus Torvalds to handle the development of Linux

- Lightweight
- Fast
- Robust
- Handles large projects

The version control database is called a "Repository"

- Can both be local and remote
- There are both command line interface and graphical interfaces to work with the repositories

Version Control – Git Terminology

master - the repository’s main branch. Depending on the work flow it is the one people work on or the one where the integration happens

clone - copies an existing git repository, normally from some remote location to your local environment.

commit - submitting files to the repository (the local one); in other VCS it is often referred to as “checkin”

fetch or pull - is like “update” or “get latest”. The difference between fetch and pull is that pull combines both, fetching the latest code from a remote repo as well as performs the merging.

push - is used to submit the code to a remote repository

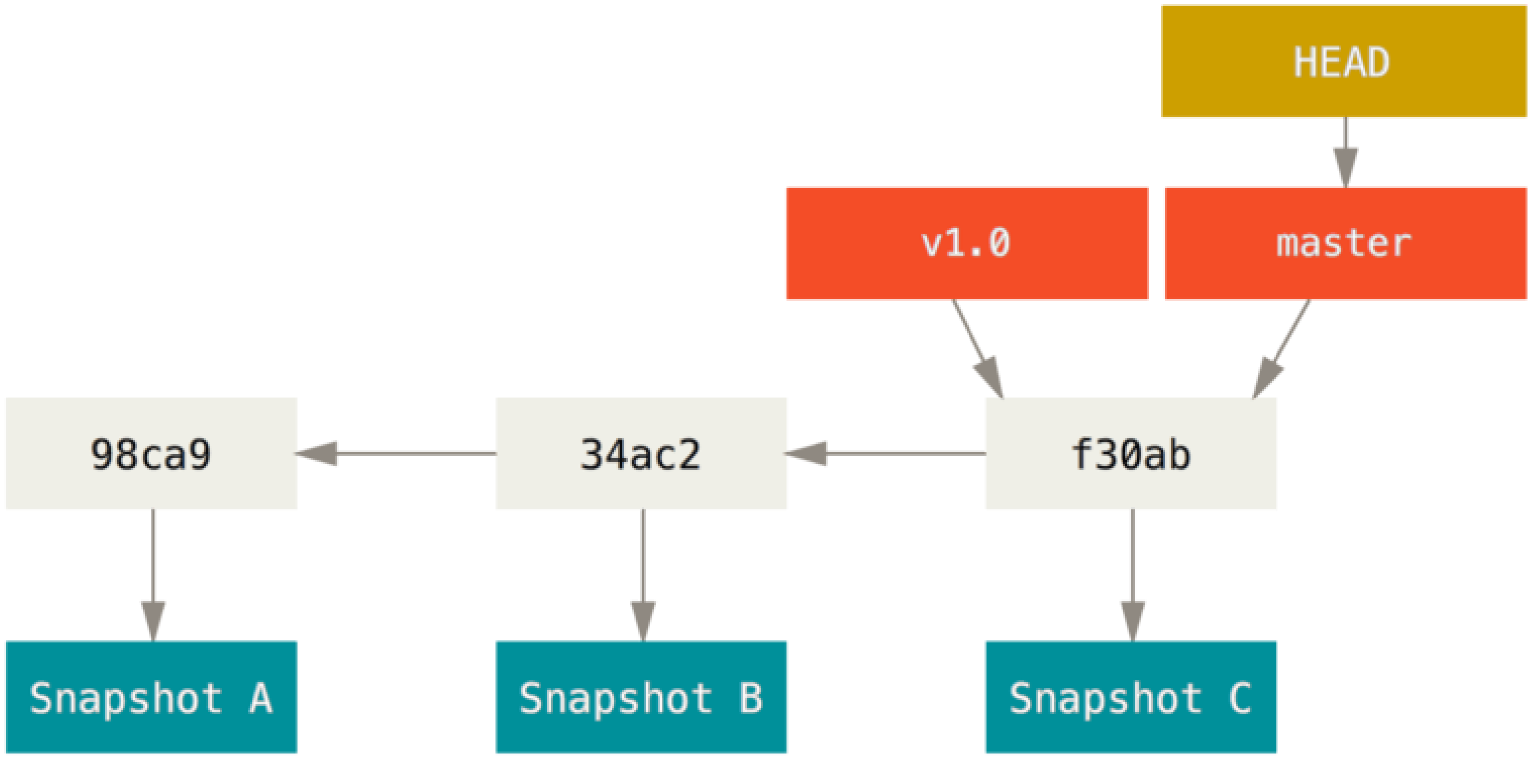
remote - these are “remote” locations of your repository, normally on some central server.

SHA - every commit or node in the Git tree is identified by a unique SHA key. You can use them in various commands in order to manipulate a specific node.

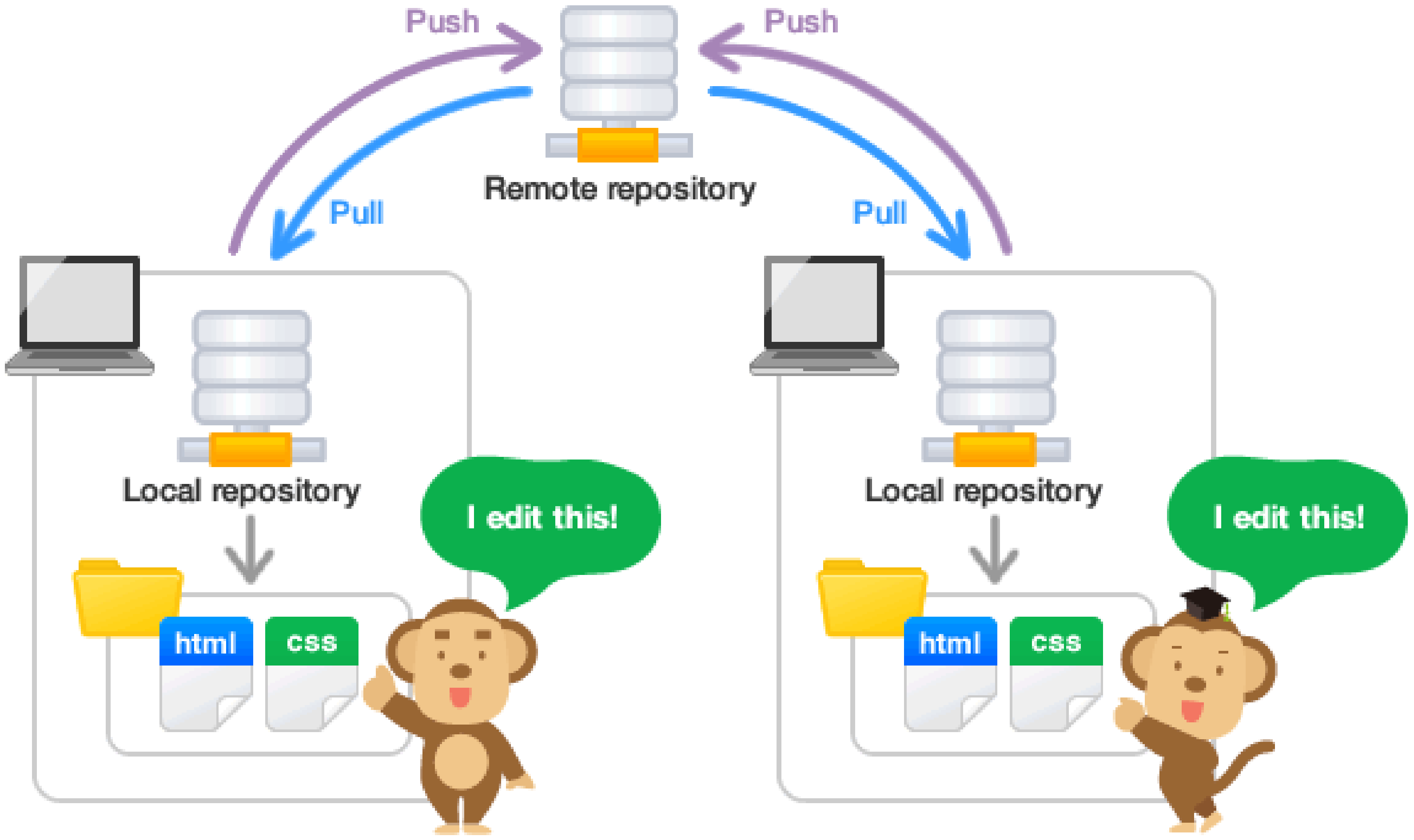
head - is a reference to the node to which our working space of the repository currently points.

branch - is just like in other VCS with the difference that a branch in Git is actually nothing more special than a particular label on a given node. It is not a physical copy of the files as in other popular VCS.

Version Control – Git



Version Control – Git



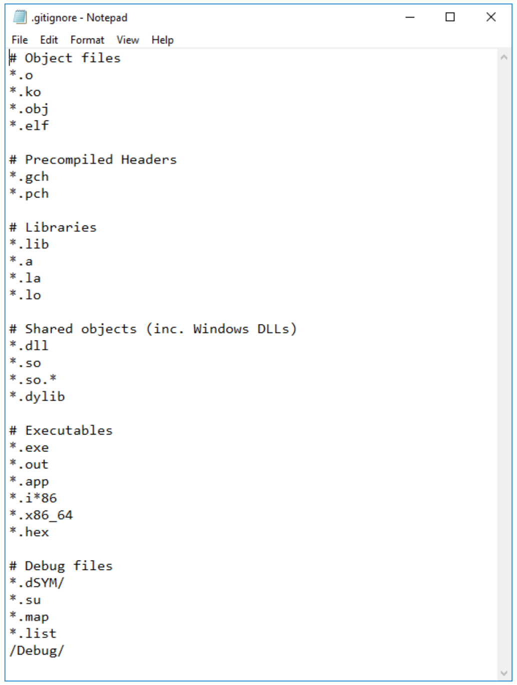
https://kevintshoemaker.github.io/StatsChats/GIT_tutorial.html

Version Control – Git ignore

Not all files in the working directory shall be subject to source control.

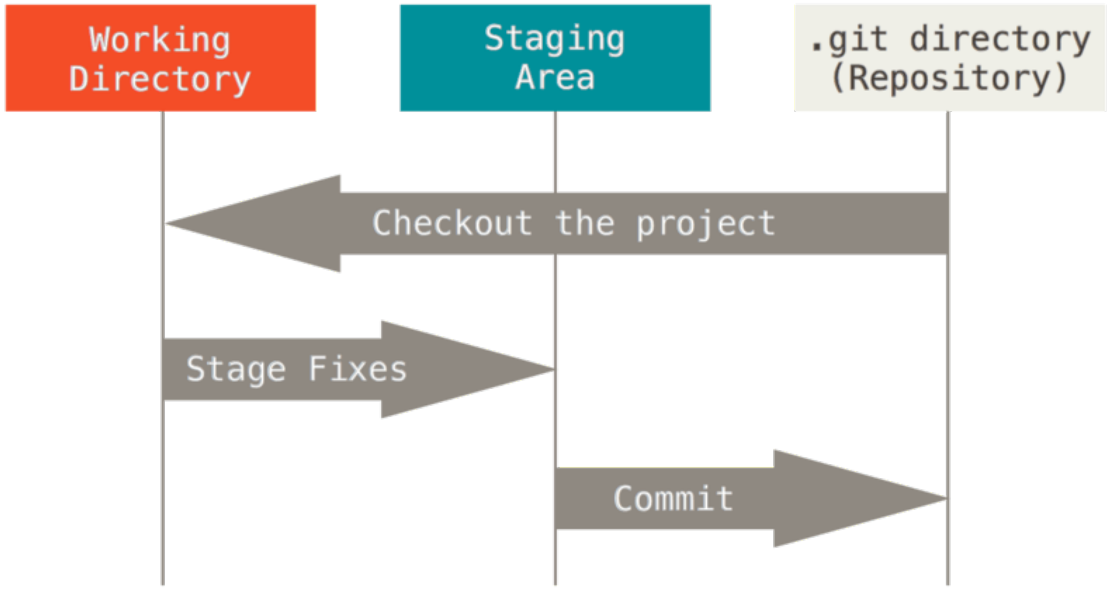
Generated artifacts such as .obj etc may not be part and should be excluded.

Git has a file called ".gitignore" which lists files that should be excluded from version control



```
# Object files
*.o
*.ko
*.obj
*.elf

# Precompiled Headers
*.gch
*.pch

# Libraries
*.lib
*.a
*.la
*.lo

# Shared objects (inc. Windows DLLs)
*.dll
*.so
*.so.*
*.dylib

# Executables
*.exe
*.out
*.app
*.i*86
*.x86_64
*.hex

# Debug files
*.dSYM/
*.su
*.map
*.list
/Debug/
```

Version Control – Git

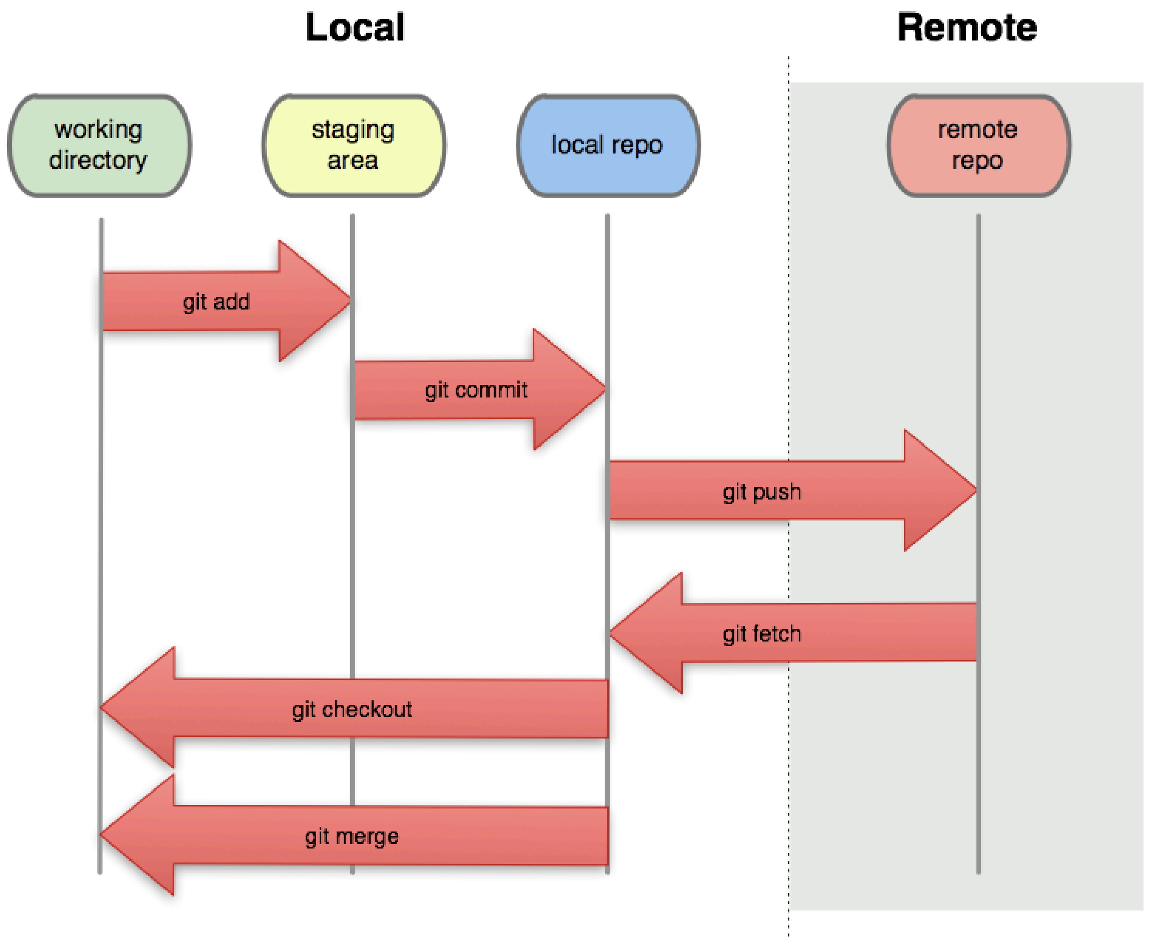


Git has three main states that your files can reside in: *committed*, *modified*, and *staged*:

Committed means that the data is safely stored in your local database.
Modified means that you have changed the file but have not committed it to your database yet.
Staged means that you have marked a modified file in its current version to go into your next commit snapshot.

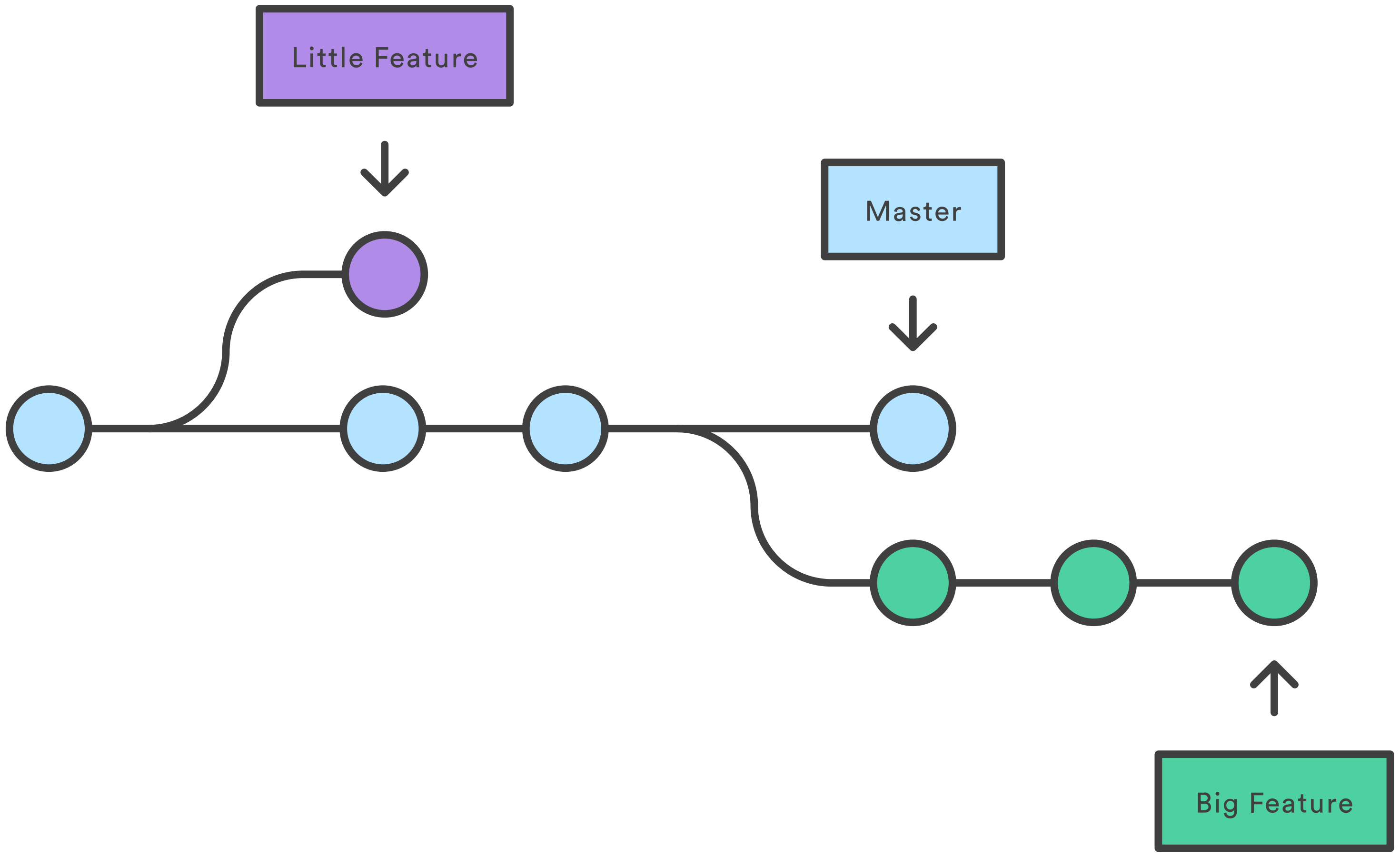
https://git-scm.com/book/en/v2/Getting-Started-Git-Basics

Version Control – Git



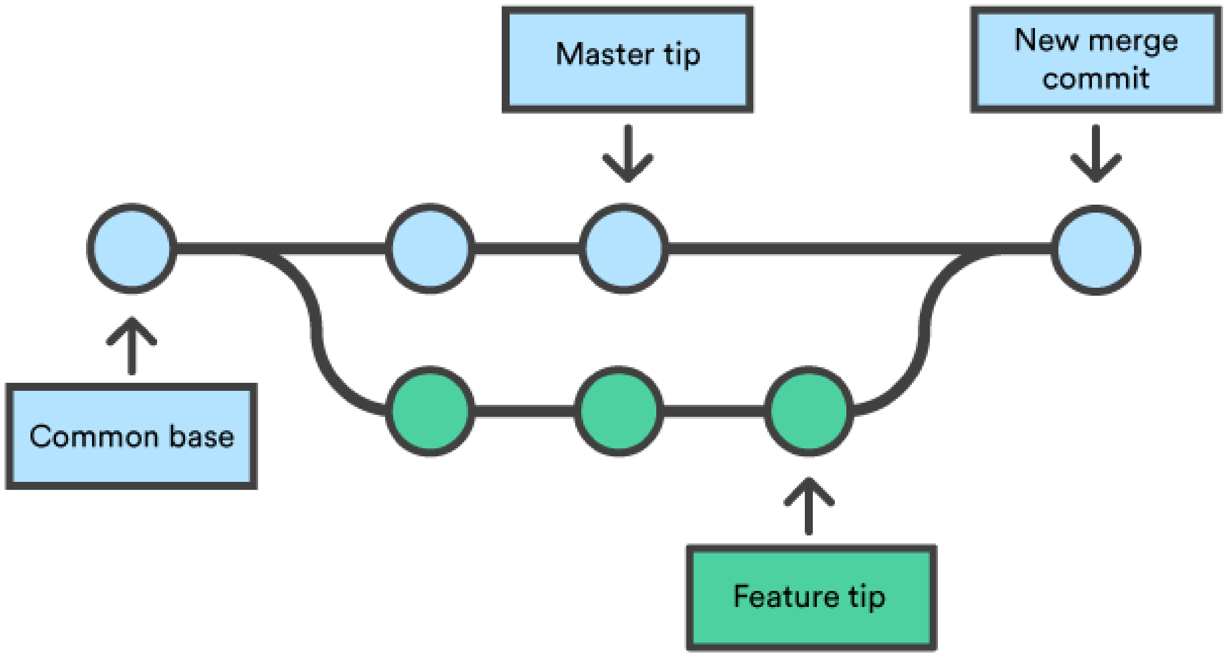
https://kevintshoemaker.github.io/StatsChats/GIT_tutorial.html

Version Control – Git branching



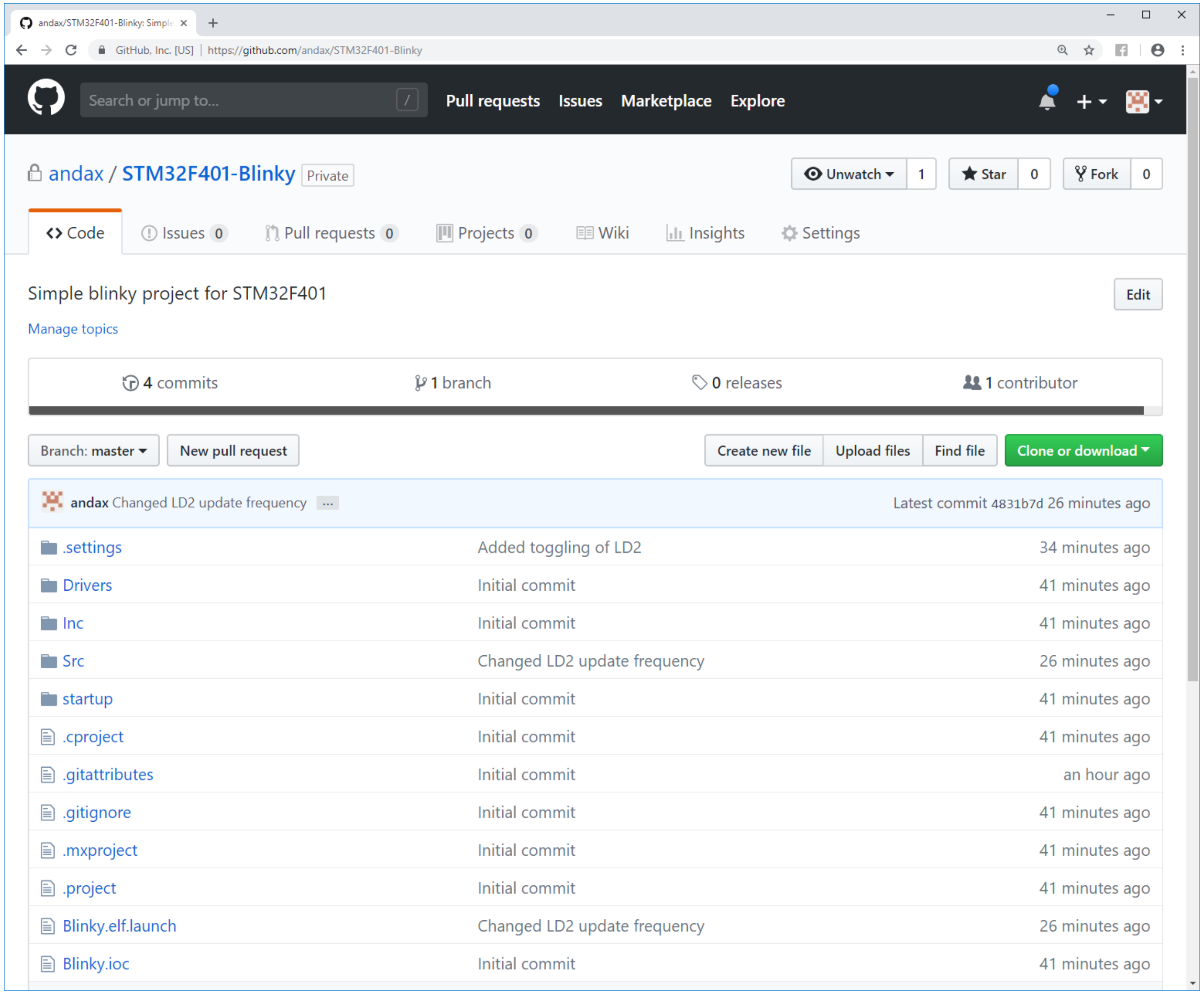
https://www.atlassian.com/git/tutorials/using-branches

Version Control – Git merging



https://www.atlassian.com/git/tutorials/using-branches/git-merge

Version Control – Github



andax/STM32F401-Blinky: Simple

GitHub, Inc. [US] | https://github.com/andax/STM32F401-Blinky

andax / STM32F401-Blinky Private

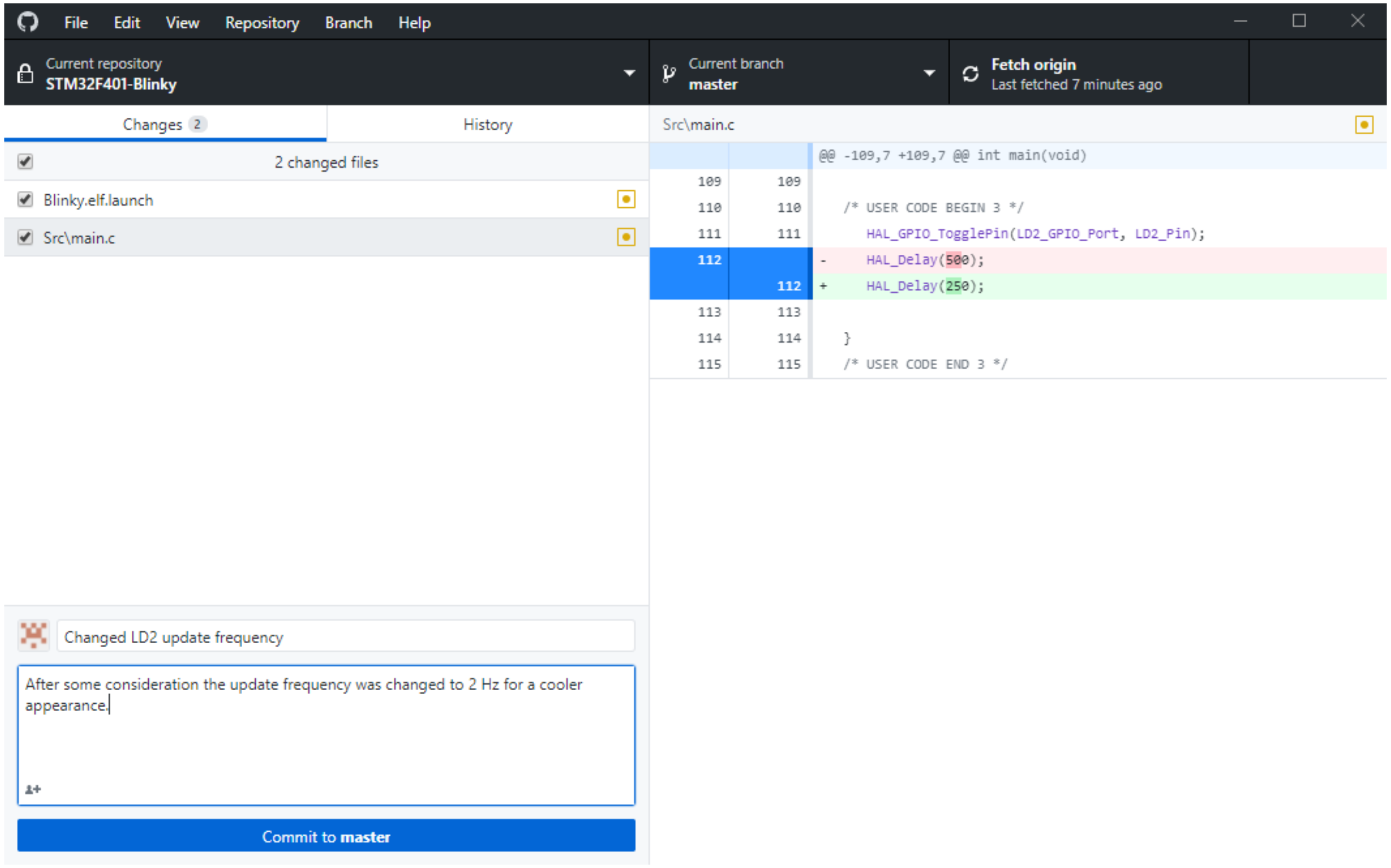
Simple blinky project for STM32F401

4 commits | 1 branch | 0 releases | 1 contributor

andax Changed LD2 update frequency ... Latest commit 4831b7d 26 minutes ago

.settings	Added toggling of LD2	34 minutes ago
Drivers	Initial commit	41 minutes ago
Inc	Initial commit	41 minutes ago
Src	Changed LD2 update frequency	26 minutes ago
startup	Initial commit	41 minutes ago
.cproject	Initial commit	41 minutes ago
.gitattributes	Initial commit	an hour ago
.gitignore	Initial commit	41 minutes ago
.mxproject	Initial commit	41 minutes ago
.project	Initial commit	41 minutes ago
Blinky.elf.launch	Changed LD2 update frequency	26 minutes ago
Blinky.ioc	Initial commit	41 minutes ago

Version Control – Github Desktop



```
@@ -109,7 +109,7 @@ int main(void)

  /* USER CODE BEGIN 3 */
    HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);
-   HAL_Delay(500);
+   HAL_Delay(250);

  }
  /* USER CODE END 3 */
```

Changed LD2 update frequency

After some consideration the update frequency was changed to 2 Hz for a cooler appearance.

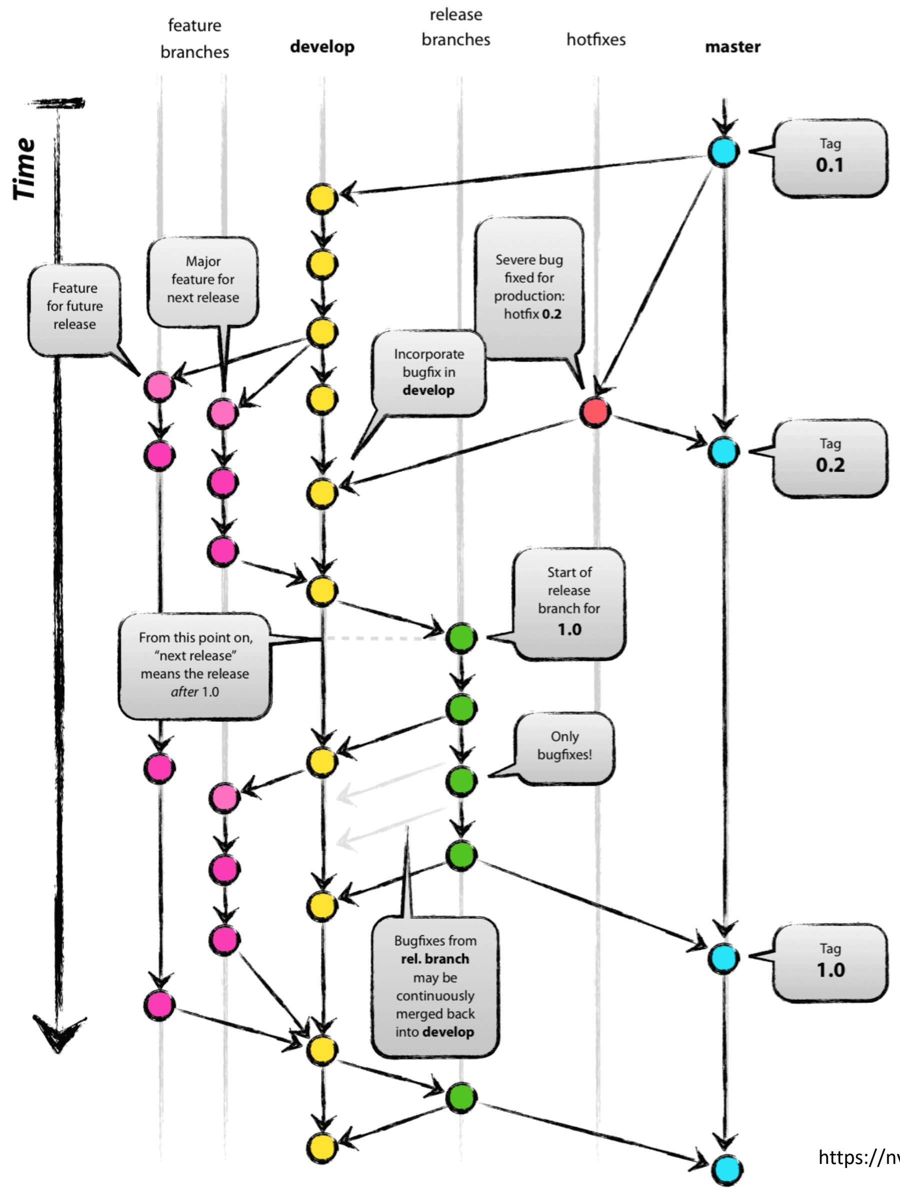
Version Control – Github

A free student account can be created at Github

https://education.github.com/pack

It gives the same features as the normal paid "Unlimited private repositories" package

Version Control – Git branching



feature branches
develop
release branches
hotfixes
master
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release after 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

https://nvie.com/posts/a-successful-git-branching-model/

Git – Terminal

```
> cd C:/somedirectory

> git init

Initialized empty Git repository in C:/somedirectory/.git/

> git add .

> git commit -m 'message’

> git remote add origin <url>

> git push –u origin master
```

<url> = https://github.com/myusername/reponame.git

Microcomputer Engineering

Questions?

Contact information

Andreas Axelsson

Email: andreas.axelsson@ju.se

Mobile: 0709-467760