Disruptive Online

Getting started with Nuttx and STM32

STM32

life.augmented

Preface

There is a lot to learn about Nuttx:

Internet:

- https://nuttx.apache.org/
- https://groups.google.com/forum/#!forum/nuttx

Youtube Channel from my friend and Nuttx mentor Alan!

- https://www.youtube.com/channel/UC0QciIlcUnjJkL5yJJBmluw/videos?app=desktop

My channel (Youtube):

- https://www.youtube.com/channel/UCKsSePEEUaRAmv2-S2x0C7w

My website:

- http://nuttx.nl/

Thank you Greg for being supportive and making Nuttx!

Who am I? Grateful to have met these guys!

Alan

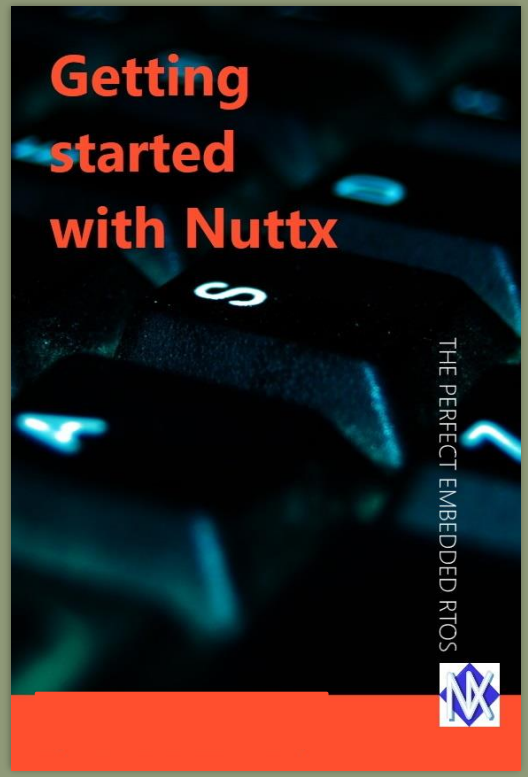
Me

Greg

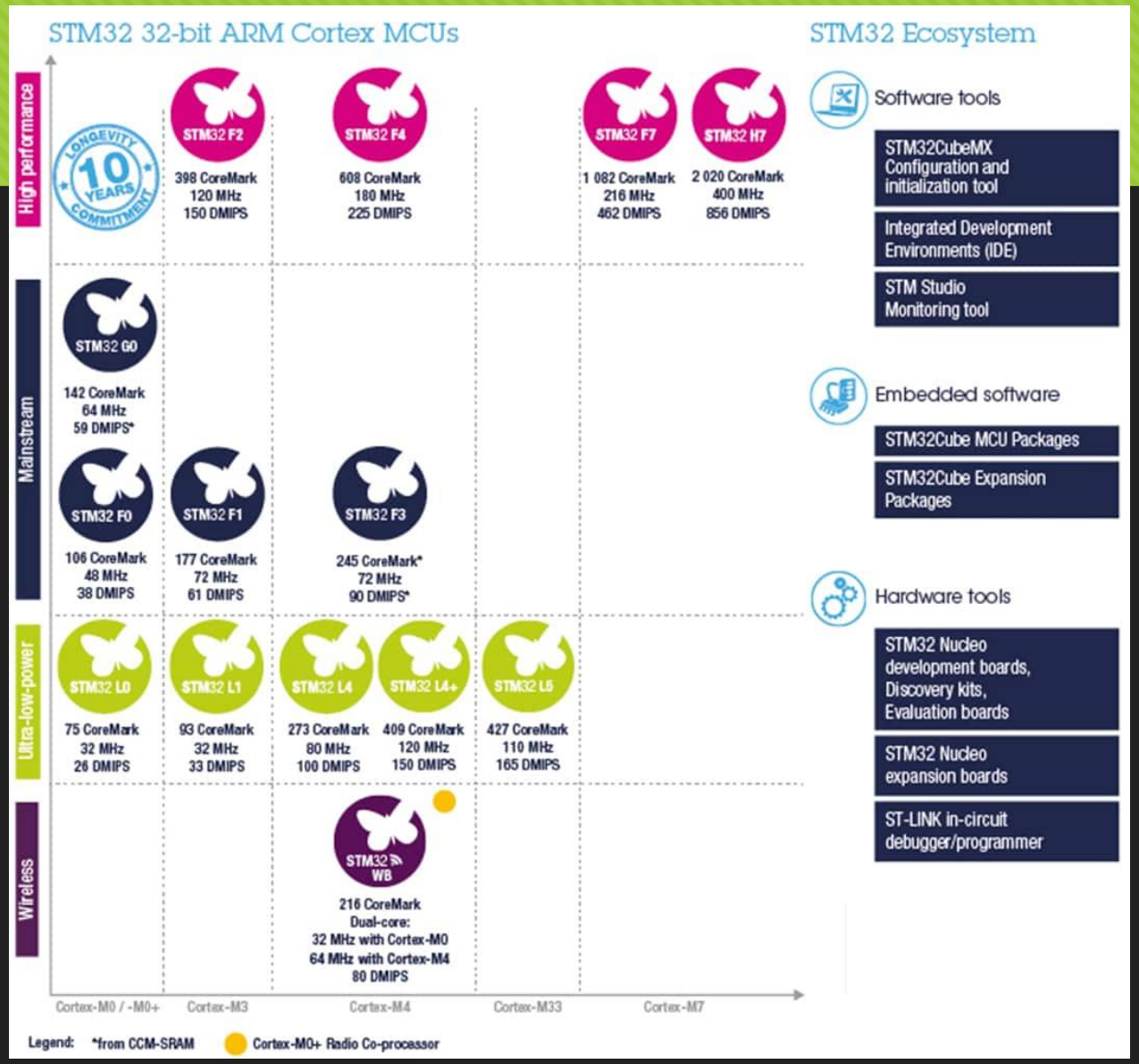
A refuge

Nuttx had received much attention when it was first released in 2007 by **Greg**ory Nutt under the permissive **BSD license**. The world is seeking an **RTOS** which can be used to address the questions which enthusiasts are in search for and want to find the most significant innovation in the world of **Internet of Things** (=IoT) and **Embedded Devices**. Many wrote books and articles in the past few years about Real-Time Operating Systems. However, if you want to learn more about how an RTOS works, you may **find yourself lost** in a universe of books that either quickly skim over the technical details or that discuss the underlying technical concepts at a highly formal level.

This **book** (**in the making**) **fills the gap** that exists between purely technical books about an RTOS, on the one hand. Also, the literature specific about its expected impact or visions and its future, on the other hand. Writing this book is **because** a **conceptual understanding** of the technical foundations of Nuttx (The Perfect Embedded RTOS) and also meant to **be a guide in your practical solutions**.



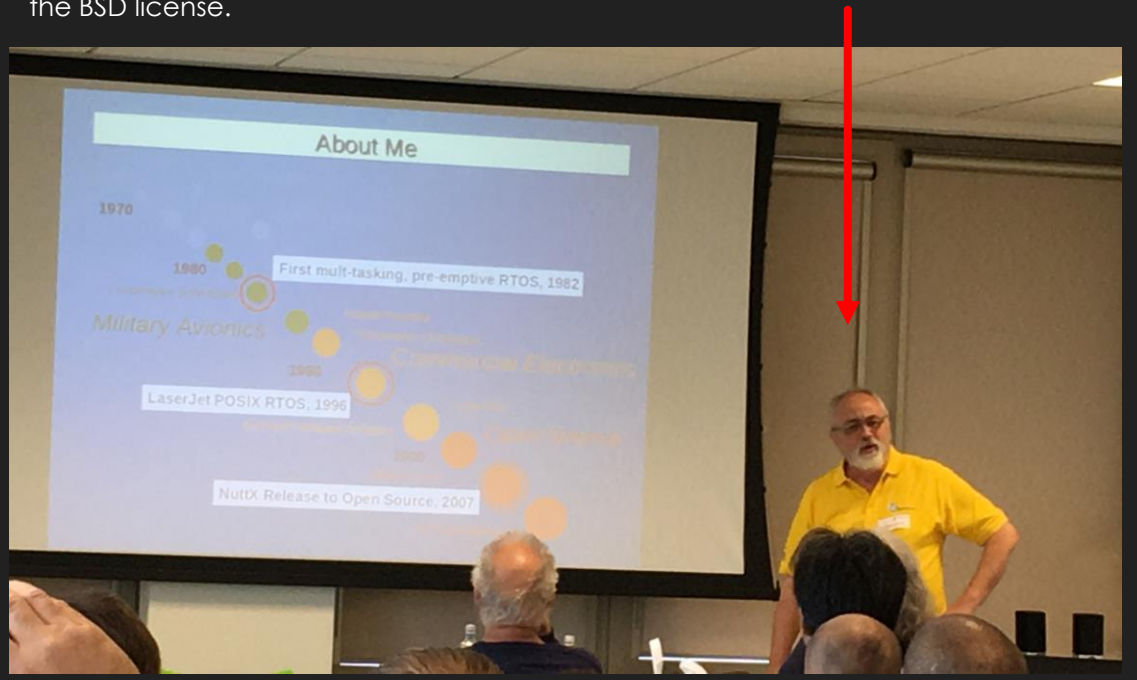
STM32


STM32 32-bit ARM Cortex MCUs
STM32 Ecosystem
High performance
Mainstream
Ultra-low-power
Wireless
LONGEVITY 10 YEARS COMMITMENT
STM32 F2
398 CoreMark
120 MHz
150 DMIPS
STM32 F4
608 CoreMark
180 MHz
225 DMIPS
STM32 F7
1 082 CoreMark
216 MHz
462 DMIPS
STM32 H7
2 020 CoreMark
400 MHz
856 DMIPS
STM32 G0
142 CoreMark
64 MHz
59 DMIPS*
STM32 F0
106 CoreMark
48 MHz
38 DMIPS
STM32 F1
177 CoreMark
72 MHz
61 DMIPS
STM32 F3
245 CoreMark*
72 MHz
90 DMIPS*
STM32 L0
75 CoreMark
32 MHz
26 DMIPS
STM32 L1
93 CoreMark
32 MHz
33 DMIPS
STM32 L4
273 CoreMark
80 MHz
100 DMIPS
STM32 L4+
409 CoreMark
120 MHz
150 DMIPS
STM32 L5
427 CoreMark
110 MHz
165 DMIPS
STM32 WB
216 CoreMark
Dual-core:
32 MHz with Cortex-M0
64 MHz with Cortex-M4
80 DMIPS
Cortex-M0 / -M0+
Cortex-M3
Cortex-M4
Cortex-M33
Cortex-M7
Legend: *from CCM-SRAM
Cortex-M0+ Radio Co-processor
Software tools
STM32CubeMX Configuration and initialization tool
Integrated Development Environments (IDE)
STM Studio Monitoring tool
Embedded software
STM32Cube MCU Packages
STM32Cube Expansion Packages
Hardware tools
STM32 Nucleo development boards, Discovery kits, Evaluation boards
STM32 Nucleo expansion boards
ST-LINK in-circuit debugger/programmer

What is it?

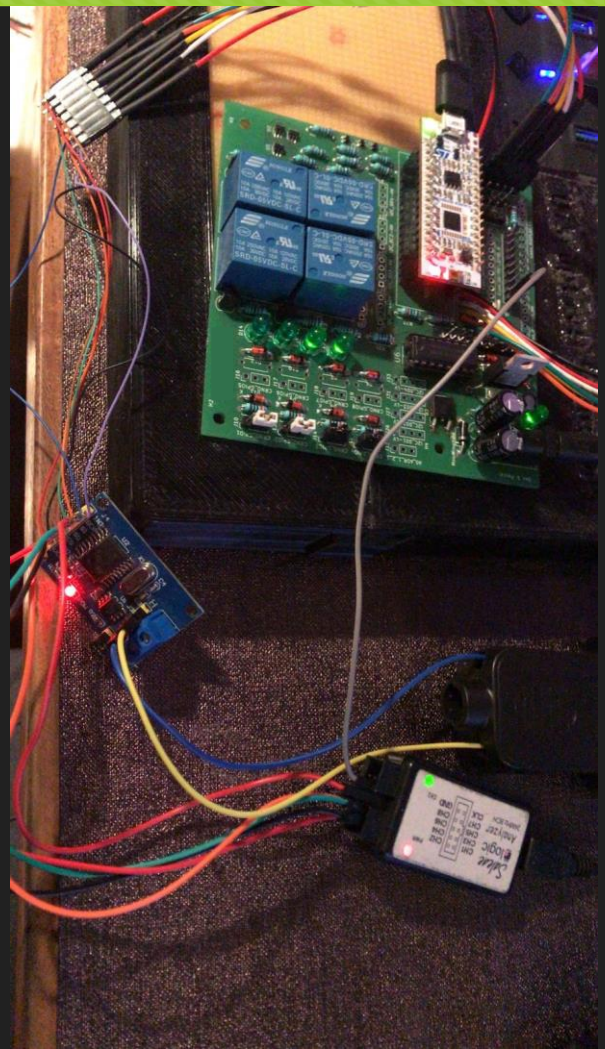
- Its an RTOS…
- It has: Performance, Real Time == Deterministic, Nuttx tasks, can handle a thread within an environment (like a Linux process),
  Each thread has its own stack,
  Each thread has an execution priority managed by the OS,
  Each thread is a member of a "task group",
  Share resources (like a Linux process),
  Can wait for events or resource availability
  Threads communicate via Interprocess Communications (IPC):,
  POSIX Message Queues, Signals, Counting semaphores, etc.,
  Standard POSIX / Linux compatible,
  NuttX supports use of standard IPCs from interrupt handlers, -
  The Nuttx kernel, - Synchronous vs Asynchronous Context Switch,
  Asynchronous Context Switch == Interrupt Context Switch, Critical part of realtime response, VERY efficient in NuttX… Near zero additional overhead,
  Synchronous Context Switch, Thread relinquishes CPU by waiting for event, NOT a critical part of realtime response, But may be important to overall performance and throughput, -
  High Priority, Zero Latency Interrupts High Priority, Zero Latency Interrupts, NuttX implements with: Higher interrupt priority, Direct vector to C code, Indirect interrupt context switches via PENDSV, - Security aspects, - (Etc..), Higher-
  half and lowerhalf : So board specific logic and architecture specific logis is situated in the "lower-half" (address ranges: 0x00000000 - 0xBFFFFFFF) and drivers and the Nuttx Core Logic is positioned in the higher-half (address ranges: 0xC0000000 - 0xFFFFFFFF)

Gregory Ellis Nutt is an American embedded software developer, author and main contributor to the NuttX real-time operating system. Between 2004 and 2007, Nutt developed the core of the NuttX operating system, and released it in 2007 under the BSD license.



Why do we need it?

- It really works!!!



Getting started!!

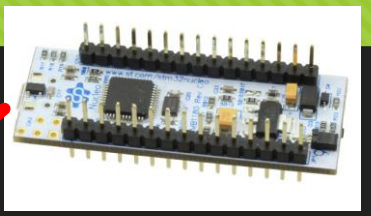
The most challenging part….

- A live session about the most important aspects concerning Nuttx and STM32…

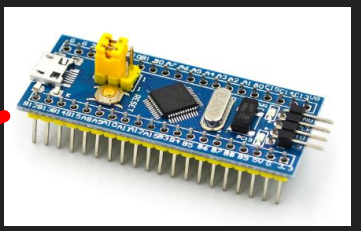
What we use part I

Debugger/Programmers

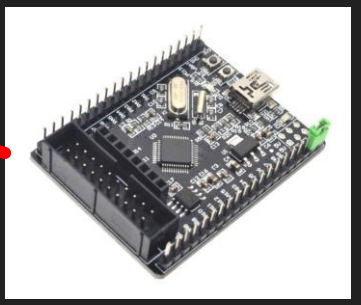
STM32L432KC Nucleo



STM32F103 Bluepill

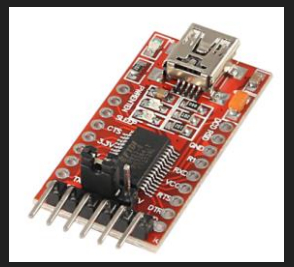


STM32F103 Smart

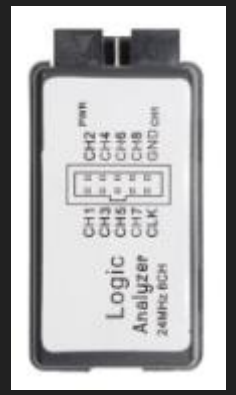


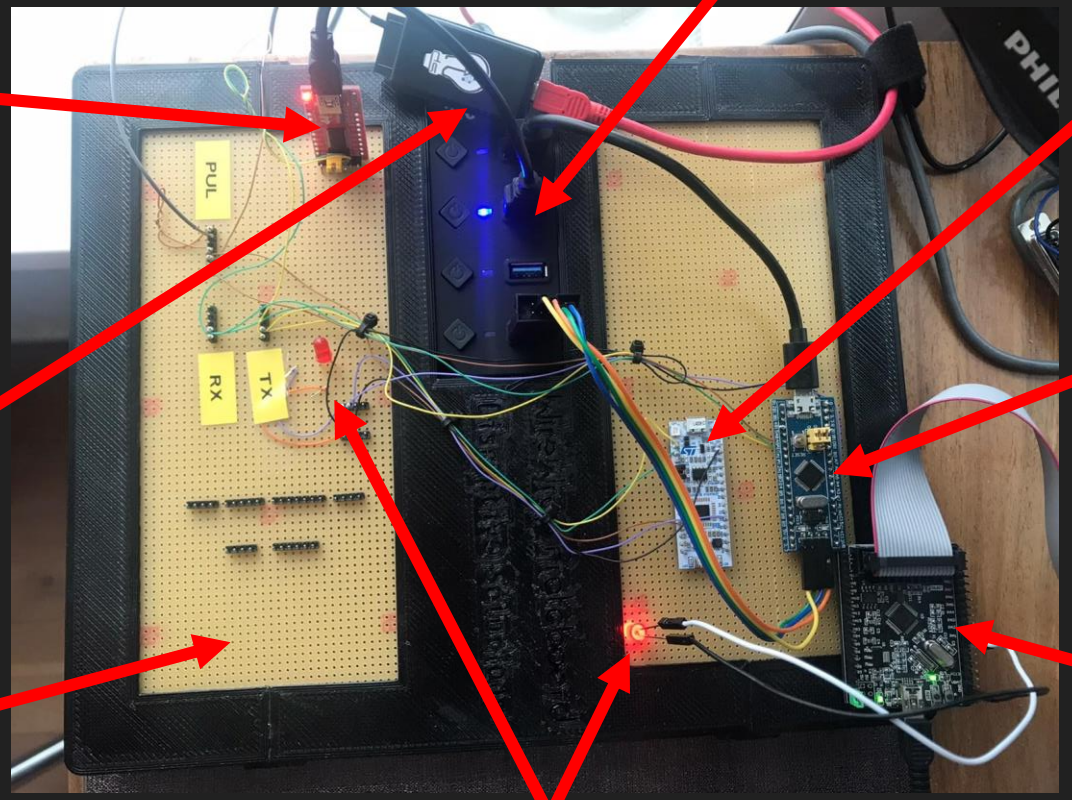
USB Hub

FTDI



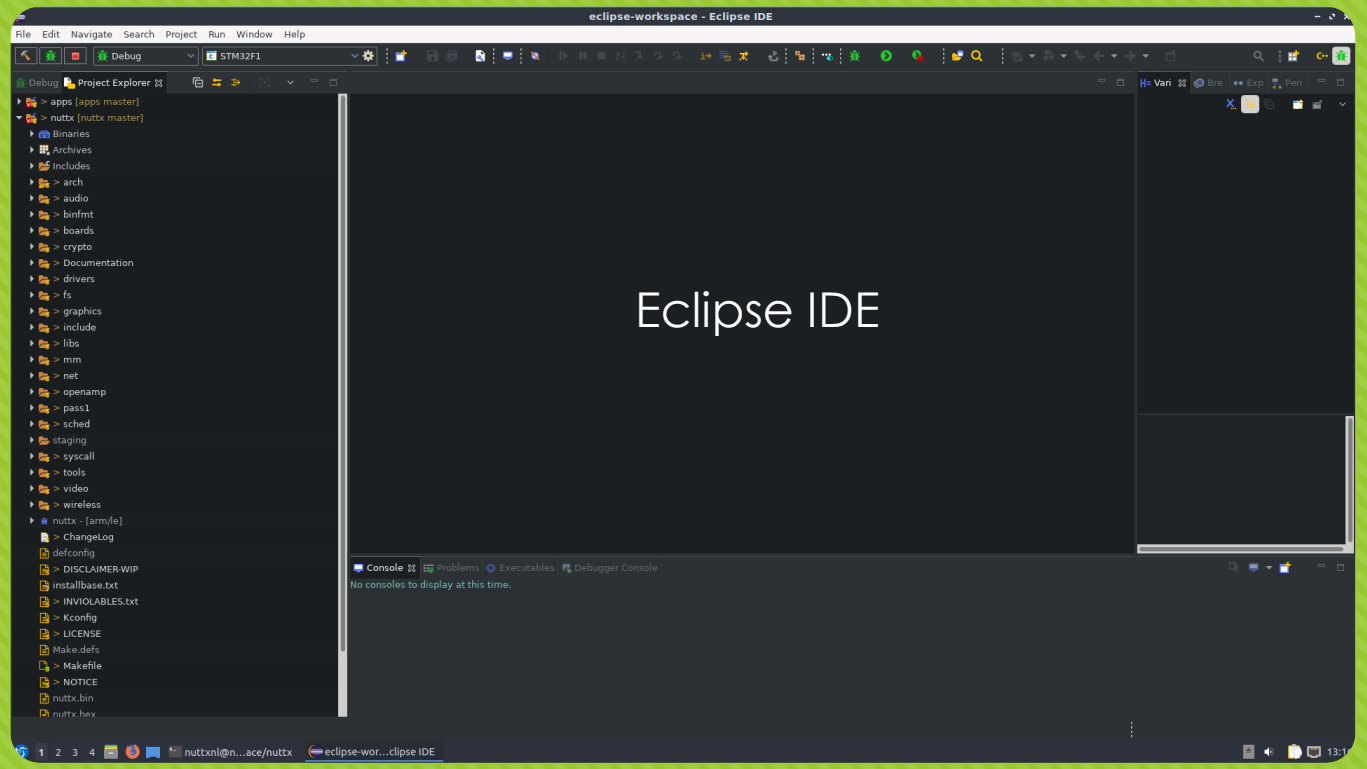
Logic Analyzer





Wire Wrap Protoboard

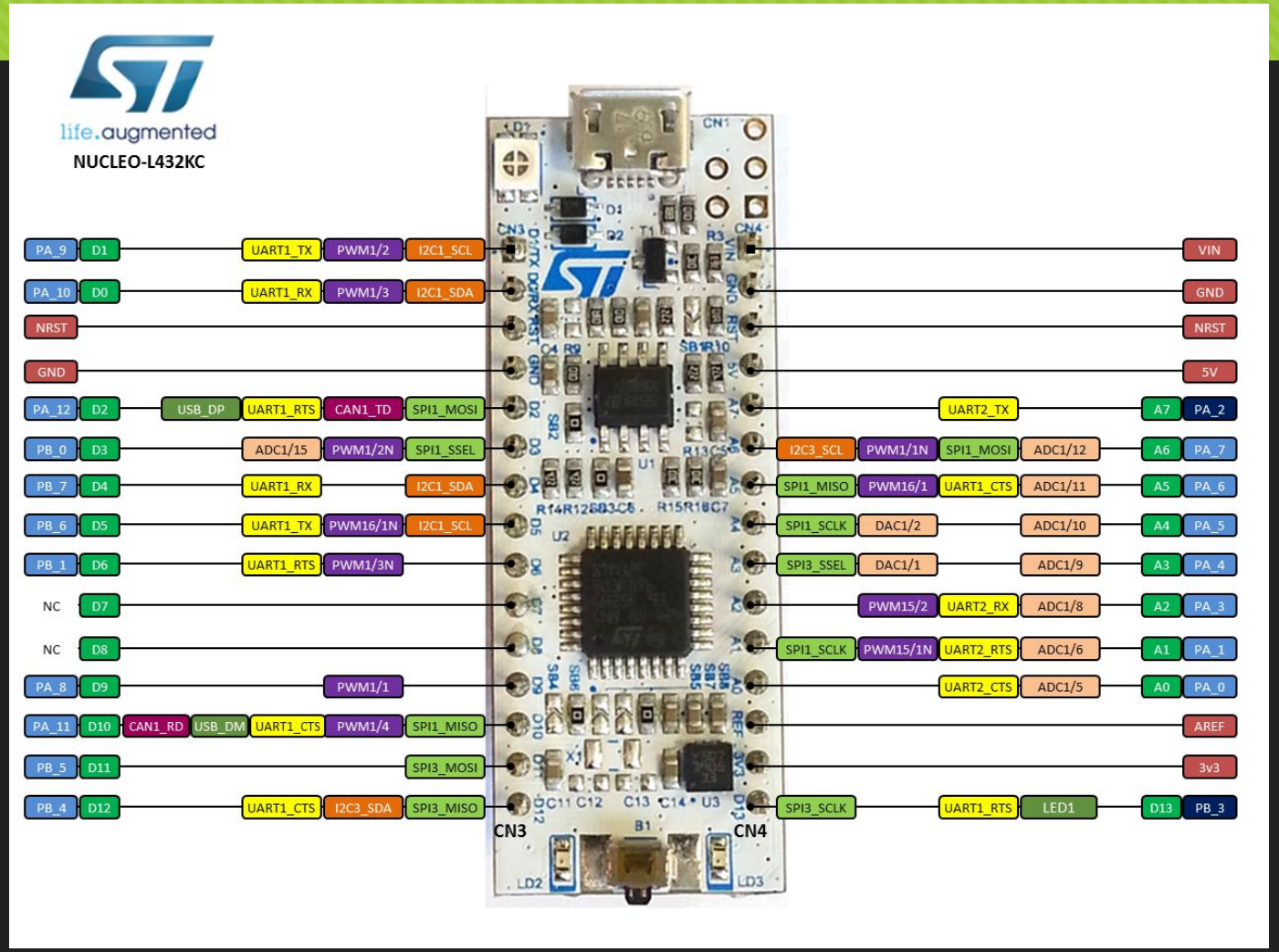
LEDS



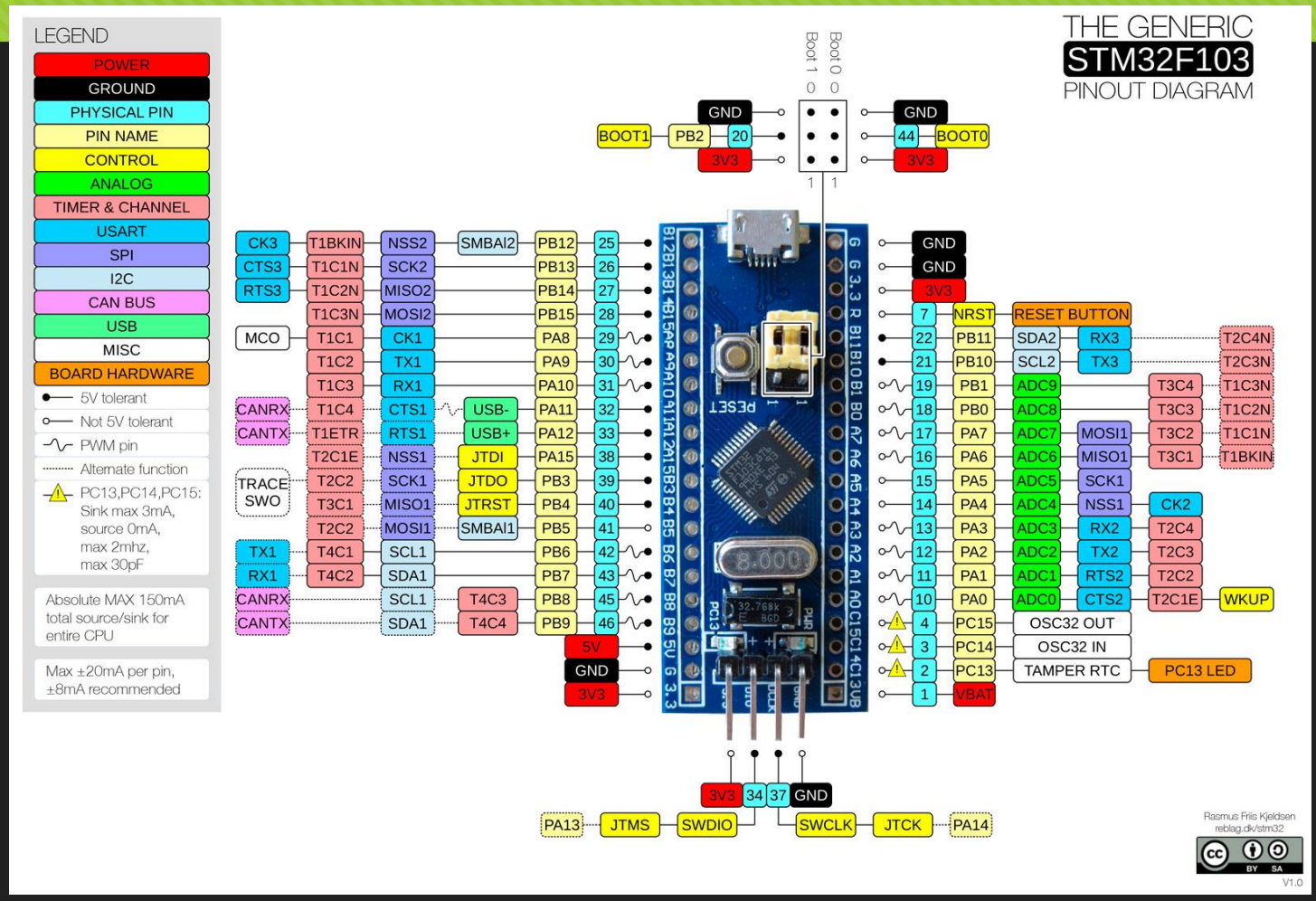
Our VirtualBox image as base

What we use part II

Pinouts


life.augmented
NUCLEO-L432KC
PA_9 D1 UART1_TX PWM1/2 I2C1_SCL
PA_10 D0 UART1_RX PWM1/3 I2C1_SDA
NRST
GND
PA_12 D2 USB_DP UART1_RTS CAN1_TD SPI1_MOSI
PB_0 D3 ADC1/15 PWM1/2N SPI1_SSEL
PB_7 D4 UART1_RX I2C1_SDA
PB_6 D5 UART1_TX PWM16/1N I2C1_SCL
PB_1 D6 UART1_RTS PWM1/3N
NC D7
NC D8
PA_8 D9 PWM1/1
PA_11 D10 CAN1_RD USB_DM UART1_CTS PWM1/4 SPI1_MISO
PB_5 D11 SPI3_MOSI
PB_4 D12 UART1_CTS I2C3_SDA SPI3_MISO
CN3
VIN
GND
NRST
5V
UART2_TX A7 PA_2
I2C3_SCL PWM1/1N SPI1_MOSI ADC1/12 A6 PA_7
SPI1_MISO PWM16/1 UART1_CTS ADC1/11 A5 PA_6
SPI1_SCLK DAC1/2 ADC1/10 A4 PA_5
SPI3_SSEL DAC1/1 ADC1/9 A3 PA_4
PWM15/2 UART2_RX ADC1/8 A2 PA_3
SPI1_SCLK PWM15/1N UART2_RTS ADC1/6 A1 PA_1
UART2_CTS ADC1/5 A0 PA_0
AREF
3v3
SPI3_SCLK UART1_RTS LED1 D13 PB_3
CN4

Pinouts


LEGEND
POWER
GROUND
PHYSICAL PIN
PIN NAME
CONTROL
ANALOG
TIMER & CHANNEL
USART
SPI
I2C
CAN BUS
USB
MISC
BOARD HARDWARE
5V tolerant
Not 5V tolerant
PWM pin
Alternate function
PC13,PC14,PC15: Sink max 3mA, source 0mA, max 2mhz, max 30pF
Absolute MAX 150mA total source/sink for entire CPU
Max ±20mA per pin, ±8mA recommended
THE GENERIC
STM32F103
PINOUT DIAGRAM
Boot 1
Boot 0
GND
BOOT1
PB2
20
3V3
44
BOOT0
CK3
CTS3
RTS3
MCO
CANRX
CANTX
TRACE SWO
TX1
RX1
CANRX
CANTX
T1BKIN
T1C1N
T1C2N
T1C3N
T1C1
T1C2
T1C3
T1C4
T1ETR
T2C1E
T2C2
T3C1
T2C2
T4C1
T4C2
NSS2
SCK2
MISO2
MOSI2
CK1
TX1
RX1
CTS1
RTS1
NSS1
SCK1
MISO1
MOSI1
SCL1
SDA1
SCL1
SDA1
SMBAI2
USB-
USB+
JTDI
JTDO
JTRST
SMBAI1
T4C3
T4C4
PB12
PB13
PB14
PB15
PA8
PA9
PA10
PA11
PA12
PA15
PB3
PB4
PB5
PB6
PB7
PB8
PB9
25
26
27
28
29
30
31
32
33
38
39
40
41
42
43
45
46
5V
GND
3V3
GND
GND
3V3
7
22
21
19
18
17
16
15
14
13
12
11
10
4
3
2
1
NRST
PB11
PB10
PB1
PB0
PA7
PA6
PA5
PA4
PA3
PA2
PA1
PA0
PC15
PC14
PC13
VBAT
RESET BUTTON
SDA2
SCL2
ADC9
ADC8
ADC7
ADC6
ADC5
ADC4
ADC3
ADC2
ADC1
ADC0
OSC32 OUT
OSC32 IN
TAMPER RTC
RX3
TX3
MOSI1
MISO1
SCK1
NSS1
RX2
TX2
RTS2
CTS2
CK2
T2C4
T2C3
T2C2
T2C1E
PC13 LED
T3C4
T3C3
T3C2
T3C1
T2C4N
T2C3N
T1C3N
T1C2N
T1C1N
T1BKIN
WKUP
3V3
34
37
GND
PA13
JTMS
SWDIO
SWCLK
JTCK
PA14
Rasmus Friis Kjeldsen
reblag.dk/stm32
V1.0