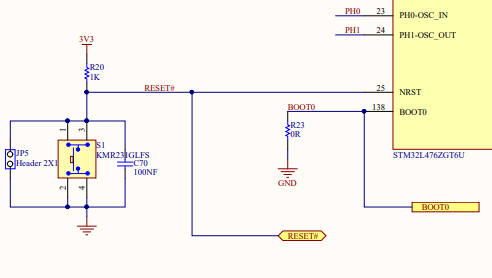
Présentation du stm32 sur le BootLoader

Accès au BOOT



Comment avoir accès au bootLoader :

Sur le STM32L476RG disposent jusqu’à 1Mo de mémoire Flash intégrée pour stocker des programmes et des données. La mémoire Flash est divisée en deux bank permettant des opérations de lecture en écriture.

Cette fonction permet d’effectuer une opération de lecture à partir d’une bank tout en une opération d’effacement ou de programmation est effectuée sur l’autre banque. La dual bank boot est prise en charge également.

Chaque bank contient 256 pages de 2 Ko.

Des protections flexibles peuvent être configurées grâce aux octets d’option :

. Protection de lecture (RDP) pour protéger toute la mémoire, 3 niveaux sont disponibles :

- Niveau 0 : pas de protection de lecture.

- Niveau 1 : protection contre la lecture de la mémoire : la mémoire Flash ne peut pas être lue depuis ou écrit si l’une des fonctionnalités de débogage est connecté, le démarrage de la RAM ou le chargeur de démarrage est choisi.

- Niveau 2 : protection contre la lecture de la puce fonctionnalités de débogage (Cortex-M4 JTAG et série fil), le démarrage dans la RAM et la sélection du chargeur de démarrage sont désactivés (fusible JTAG) Cette sélection est irréversible.

Les appareils STM32L476xx comportent jusqu'à 128 Ko de SRAM intégrée. Cette SRAM est divisée en deux blocs: • 96 Ko mappés à l'adresse 0x2000 0000 (SRAM1) • 32 Ko situés à l'adresse 0x1000 0000 avec contrôle de parité matérielle (SRAM2). Ce bloc est accessible via les bus ICode/DCode pour des performances maximales. Ces 32 Ko de SRAM peuvent également être conservés en mode veille. La SRAM2 peut être protégée en écriture avec une granularité de 1 Ko. La mémoire est accessible en lecture/écriture à la vitesse d'horloge du processeur avec 0 états d'attente

L’appareil intègre un pare-feu qui protège les données sensibles et sécurisées du code de tout accès effectué par un code exécuté en dehors des zones protégés.

3 segments peuvent être protégés et définis aux registres Firewall.

BOOT MODES

Au démarrage la broche BOOT0 et le bit d’option BOOT1 sont utilisés pour sélectionner l’une des 3 options de démarrage :

. Démarrage à partir de l’utilisateur Flash.

. Démarrage à partir de la mémoire système.

. Démarrage à partir de la SRAM intégrée.

La chargeur de démarrage est situé dans la mémoire système. Il est utilisé pour reprogrammer la mémoire Flash en utilisant USART, I2C, SPI, CAN, ou USB OTG FS en mode appareil via DFU (appareil mise à jour de fimware)

connecSens : BOOTLOADER (programme C)

la carte prototype est flashée avec le bootloader 1.0.1 DEBUG

Fonctionnalités de 1.0.1

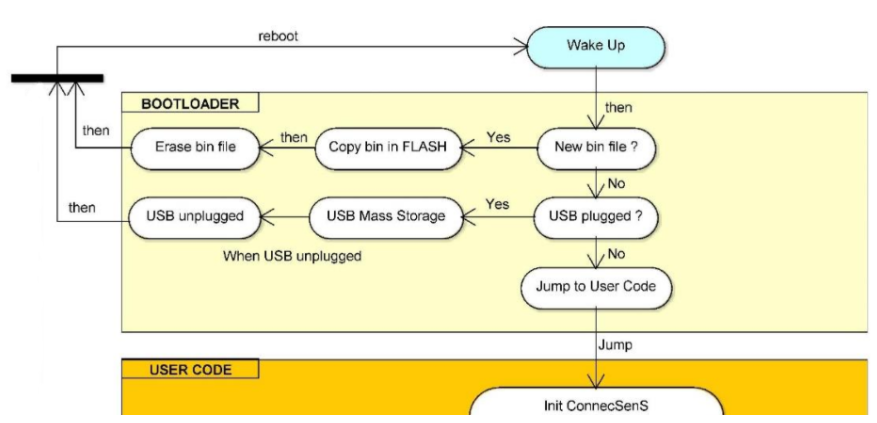
. Gestion de l’USB pour l’affichage du contenu de la carte SD lorsque la carte microcontrôleur est connecté.

. Gestion des mises à jour : lorsque la carte microcontrôleur détecte un fichier FIRMWARE.BIN (en majuscule), elle installe le nouveau programme puis redémarre l’installation d’un fichier de mise à jour environ 15 secondes.

. DEBUG : Dans cette version, le Bootloader ne supprime pas le fichier /env/state.json sur la carte SD lors d’une connexion USB ou lors d’une mise à jour. Ce fichier sert à observer la dernière heure de réveil du système et les états des périodes des périphériques. Dans la version finale du Bootloader, ce fichier sera supprimé à chaque mise à jour et à chaque connexion USB pour maintenir le système dans un état stable (redémarrage de tous les modules à chaque intervention d’un opérateur)

Le chargeur de démarrage (bootloader) est toujours le premier programme appelé. Il n’est pas possible de changer le chargeur de démarrage, sauf à utiliser la sonde de débogage.

Le bootloader s’occupe de mettre à jour le micrologiciel applicatif, présenter le nœud comme un disque USB si le câble USB est connecté et lancer l’application.



Acces au Bootloader de la nucleoL476RG

Le démarrage à partir de la mémoire flash du système entraîne l’exécution du code du bootloader stocké dans le système, la mémoire flash est protégé contre l’écriture et l’effacement cela permet la programmation dans le système (ISP)

C’est-à-dire en flashant la mémoire flash de l’utilisateur STM32. Il permet également d’écrire des données dans la RAM. Les données entrant via l’une des interfaces de communication telle que USART, SPI, I2C, USB, CAN.

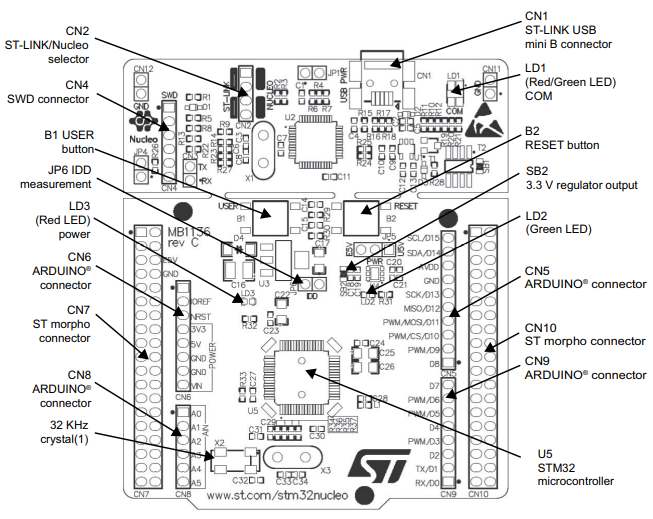
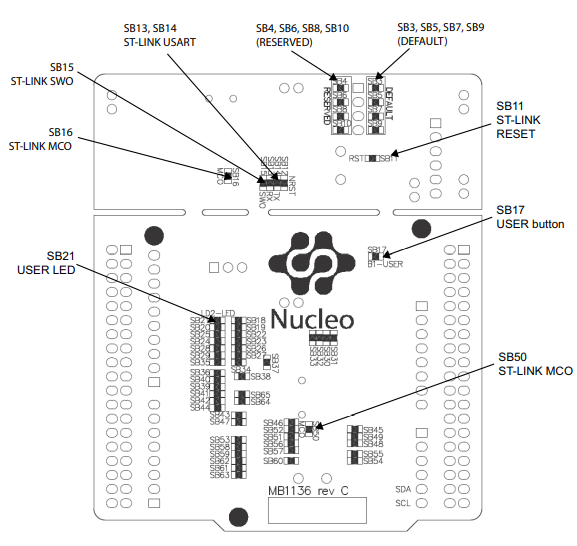
La version du bootloader peut être identifiée en lisant l’ID du bootloader à l’adresse **0x1FFF6FFE**

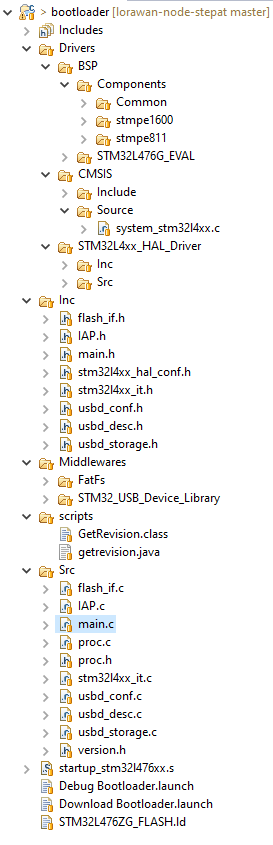
La partie soudé STM32L476RGT6 sur la nuclé est marquée d’un code date correspond à la date de fabrication et avant ou égal à la semaine 22 de 2015 équipé du bootloader V9.0 affecté par le la limitation à contourner et pièce commençant semaine 23 de 2015 contiennent le boot V9.2 où la limitation n’existe plus.

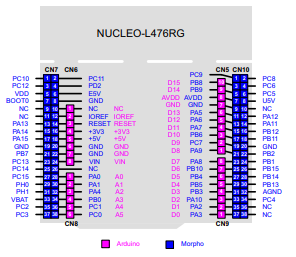
Si données corrompues télécharger le boot STSW-STM32158 package de correctif V.9.0 sur [www.st.com](http://www.st.com) et chargez le patch bootloader V9.0SARM sinon si c’est CAN charger le.

<https://github.com/STMicroelectronics>

<https://github.com/theapi/stm32/blob/master/kitchen_timer/Drivers/STM32L0xx_HAL_Driver/Inc/stm32l0xx_hal_flash.h>





<https://www.st.com/resource/en/application_note/cd00167594-stm32-microcontroller-system-memory-boot-mode-stmicroelectronics.pdf#page331>

<https://www.st.com/en/embedded-software/stm32cubel4.html>

Cavalier CN7 (5-7Boot0) à 1 accès au boot

Accès au boot avec l’usb

Accès au boot avec Emulateur STLinkV2

La plupart des STM32 sont équipés d’un bootloader hardware (non écrasable) qui permet de flasher la puce sans émulateur écrit dans la note AN2606 :STM32 microcontroller système memory boot mode

Emulateur c’est un appareil pour dialoguer avec le cœur à travers une liaison de débug ARM Cortex-M, il s disposent d’un bus SWD permettant d’accéder aux fonctions de programmation et débugage avec un nombre de broche réduit. Nous avons besoin de sortir la broche SWDIO et SWCLK, ainsi que les signaux VDD, VSS et RST sur un connecteur c’est tout. Et un émulateur STlinkV2 il est embarqué sur notre carte nucleo.

Si toute les conditions sont réunies il est possible de flasher directement par usb mai l’autre offre plus de possibilité.

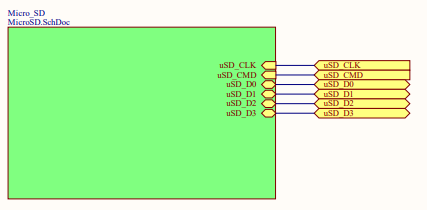
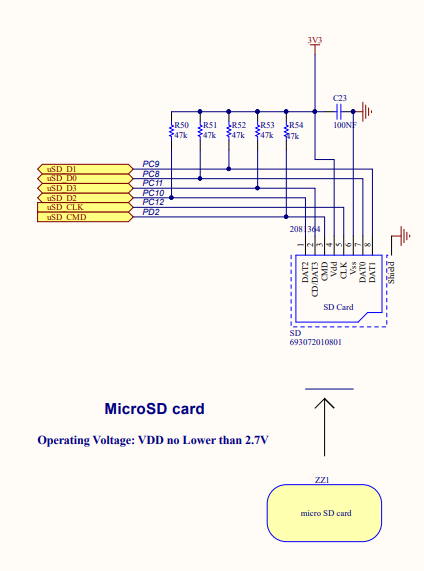
Présence de la cible :

Vérifions que le microcontrôleur est bien passé en mode bootloader.

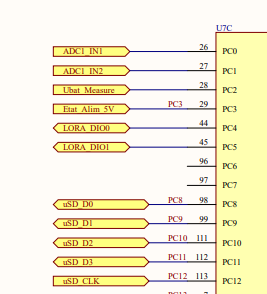
<https://haum.org/stm32_flashing.html>

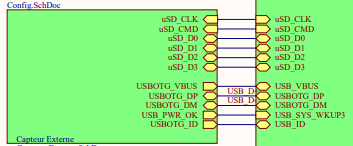
Schématisation

Carte Micro SD













Hield SD card 693072010801

PIN

usD\_D0 PC8 (98)

usD\_D1 PC9 (99)

usD\_D2 PC10 (111)

usD\_D3 PC11 (112)

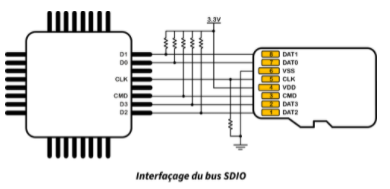
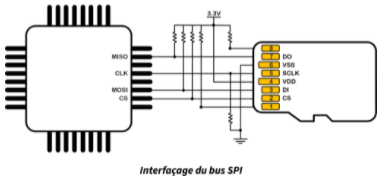
usD\_CLK PC12 (113)

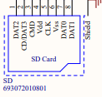
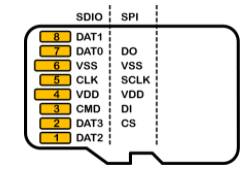
usD\_CMD PD2 (116)

SD Detect PA8 (100)

Carte SD comment ça marche ?

SDIO ou SPI, le protocole SD utilise des lignes séparées pour les commandes et les données, tandis que SPI utilise une seule paire de fils pour les deux. Ainsi dans certaines situations SDIO 1 bit peut être plus rapide que SPI, car il est possible d’envoyer des commandes en même temps que des données. Mais de l’autre le protocole SPI est documenté publiquement donc plus facile, de plus seuls quelques-uns prennent en charge nativement SD mais beaucoup font le SPI un meilleur choix pour les amateurs.





Le SDIO pour ceux avec des ressources le bus sdio offre une communication bien plus rapide grâce à un protocole qui peut supporter jusqu’à 4 lignes de données simultanées et peuvent atteindre des débits en lecture et écriture frôlant la centaine de méga-octets par seconde.

La cartes micro-SD utilisent les mêmes pattes physiques sur SPI ou SDIO le choix dépend du microcontrôleur.

Le choix du bus a un impact sur le hardware puisque le routage du bus SPI ou des bus SDIO 1bit et 4 bits différent. En plus de la connexion directe e/s, il peut s’avérer crucial d’y adjoindre des résistances de rappel ou de tirage (pull-down ou pull-up) pour s’assurer du bon fonctionnement de la carte sur SDIO ou SPI cela évite de laisser flottantes si non utilisé par le montage ou lors du boot cela est guidé par l’état au repos du signal (état haut pour D0, DAT1, DAT2, CS, DI état bas pour l’horloge CLK. Les 10Kohms à 100kohms conviendrons.

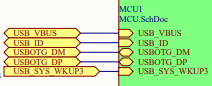
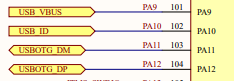
Conserver le CS ou DO, noter que l’ajout d’une pull-up CS prive de sa fonction secondaire de détection de présence de carte dans ce cas on peut utiliser un connecteur qui possède une fonction mécanique de détection avec une pull-up encore.

Le problème de consommation

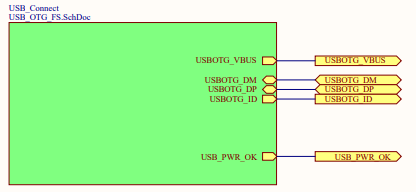
L’utilisation d’une SD carte lors de l’écriture nous avons des pic de 10 milliampères voire centaine de milliampères. Il ne faut pas avoir trop d’étage de circuit intégrés pour la gestion de l’alimentation. Une possibilité simple se tourner vers des régulateur LDO à très faible consommation délivrant un courant max de 150mA pour assurer le bon fonctionnement de la carte SD AP7350 (diode incorporated) permet de délivrer 150mA et consomme 25 microA et un courant de standby de 20nA et tout petit.

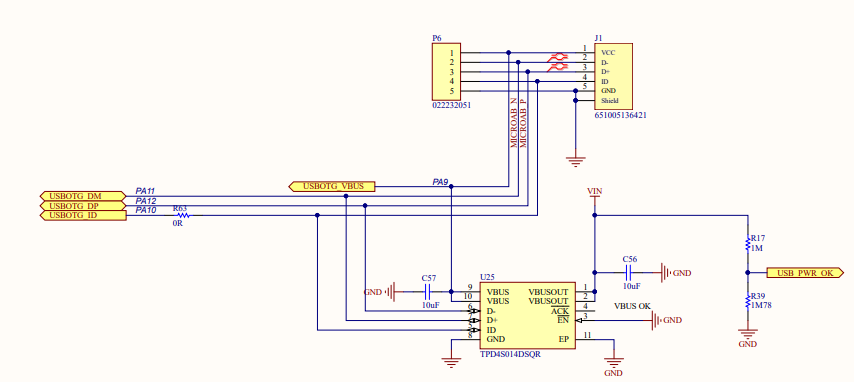
Attention tout dépend de la carte SD elle peuvent consommer moins comme les SD industrielle mais plus chère.

Carte USB











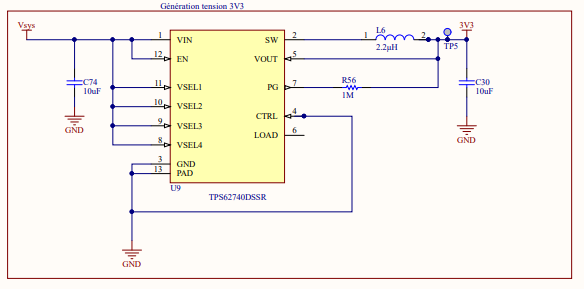
Carte Led d’informations

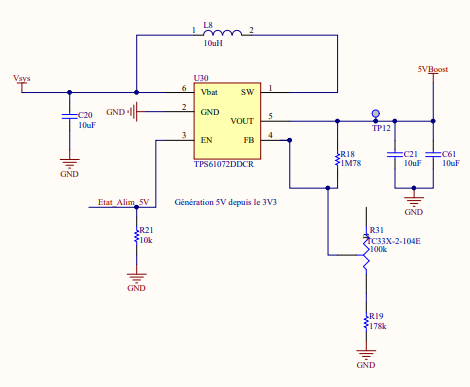


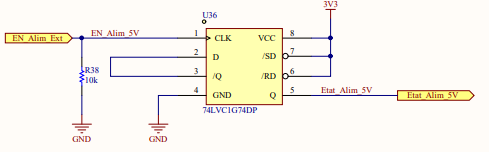












Code PIN stm32

Led1 GPIOE GPIO\_PIN\_14

Led1 GPIOE GPIO\_PIN\_3

Led2 GPIOE GPIO\_PIN\_13

Led2 GPIOE GPIO\_PIN\_4

USB GPIOA GPIO\_PIN\_9

USB GPIOA GPIO\_PIN\_10

USB GPIOA GPIO\_PIN\_11

USB GPIOA GPIO\_PIN\_12

STM32

**STM32Cube** est une initiative originale de STMicroelectronics pour faciliter la vie des développeurs en réduisant les efforts, le temps et les coûts.

**STM32Cube** couvre l'ensemble du portefeuille de produits STM32. Il comprend une plate-forme logicielle embarquée complète livrée pour chaque série STM32.

* Les modules CMSIS (cœur et périphérique) correspondant au cœur ARM(tm) implémenté dans ce produit STM32.
* Les pilotes STM32 HAL-LL, une couche d'abstraction offrant un ensemble d'API garantissant une portabilité maximisée sur l'ensemble du portefeuille STM32.
* Les pilotes BSP de chaque carte d'évaluation, de démonstration ou nucléo fournie pour cette série STM32.
* Un ensemble cohérent de bibliothèques middleware telles que RTOS, USB, FatFS, graphiques, bibliothèque de détection tactile...
* Un ensemble complet de projets logiciels (exemples de base, applications et démonstrations) pour chaque carte fournie pour cette série STM32.

Deux modèles de publication sont proposés pour le logiciel embarqué STM32Cube :

* Le **package MCU** monolithique : tous les modules logiciels STM32Cube d'une série STM32 sont présents (Drivers, Middleware, Projects, Utilities) dans le référentiel (nom usuel **STM32Cubexx** , xx correspondant à la série STM32).
* Le **composant MCU** : chaque module logiciel STM32Cube faisant partie du package STM32Cube MCU, est livré sous forme de référentiel individuel, permettant à l'utilisateur de sélectionner et d'obtenir uniquement les fonctions logicielles requises.