GIF-3002 Présentation du STM32F407 et Design matériel de SMI

Ce document présente sommairement le STM32F407 qui est le microcontrôleur utilisé dans les laboratoires du cours.

Ce document présente aussi sommairement le design de circuits avec microcontrôleurs. Il s'agit essentiellement d'une introduction aux cours qui suivront.

1 Documents reliés aux microcontrôleurs

Plusieurs documents accompagnent les microcontrôleurs modernes : une datasheet, un « programmer's manual », la documentation reliée au cœur, des applications notes et plus.

La datasheet décrit essentiellement les parties électroniques du microcontrôleur. Elle présente ses principales caractéristiques et détaille les aspects matériels et mécaniques reliés au circuit intégré.

Le « programmer's manual » ou équivalent (différents manufacturiers ont différents noms!) contient l'ensemble des informations nécessaires à la programmation du microcontrôleur. Ce document souvent très volumineux décrira chaque registre de chaque composante du microcontrôleur.

La documentation reliée au cœur est souvent indépendante du programmer's manual (voir ci-dessous pour explication). Cette documentation donne des détails sur la vitesse d'exécution des instructions et permet de coder en assembleur si nécessaire.

Enfin, des applications notes présentent des applications du microcontrôleur. Il s'agit d'exemples souvent très utiles lors de la conception d'un produit.

2 Datasheet du STM32F407

La première page de la datasheet du STM32F407 (voir DM00037051.pdf sur le site de ST Micro) présente ses caractéristiques générales :

Un cœur ARM ® Cortex ™-M4

- ***Avec accélérateur d'accès à la FLASH (ART)***
- ***Opère jusqu'à 168MHz, 210 DMIPS ou 1.25 DMIPS/MHz (Dhrystone 2.1)***
- ***Supporte des instructions DSP***

Mémoire intégrée au circuit (On-Chip memory)

- ***Jusqu'à 1MByte de FLASH***
- ***Jusqu'à 192KByte+4KByte de RAM dont 64KByte couplé avec le cœur***

Contrôleur de mémoire statique flexible

- ***Supportant la FLASH Compacte, la SRAM, la PSRAM, la NOR FLASH, et la NAND FLASH***

Interface Parallèle de LCD, supportant les modes Intel 8080 et Motorola 6800

Contrôle d'horloge de reset et d'alimentation
- ***Alimentation de 1.8V à 3.6V pour l'application (mémoires) et les I/Os***
- ***POR, PDR, PVD et BOR***
- ***Oscillateur externe 4-à-26MHz***
- ***Oscillateur RC interne 16MHz calibré en usine***
- ***Oscillateur externe 32KHz pour RTC***
- ***Oscillateur RC interne 32KHz pour RTC calibré en usine***

Basse Puissance
- ***Modes de sommeil, d'arrêt et de standby***
- ***Alimentation sur batterie pour le RTC, les registres de backup et la mémoire SRAM 4Kbyte de backup***

Trois ADCs 12 bits opérant à 2.4MSPS ou 7.2MSPS lorsque entrelacés ensemble

Deux DACs 12bits

DMA
- ***Contrôleur 16 canaux avec FIFOs et support pour les bursts de données.***

Jusqu'à 17 minuteries
- ***Jusqu'à 12 minuteries 16-bits,***
- ***2 minuteries 32-bits***
- ***Opère en compteur interne, compteur d'horloge, PWM, en compteur de pulse, en quadrature.***

Modes de debug
- ***SWD et JTAG***
- ***Cortex-M4 Embedded Trace Macrocell***

Jusqu'à 140 broches d'entrées/sorties pouvant causer des interruptions
- ***Jusqu'à 136 broches d'entrées/sorties rapides jusqu'à 84MHz***
- ***Jusqu'à 138 broches d'entrées/sorties tolérant le 5V***

Jusqu'à 15 interfaces de communication
- ***3 interfaces I2C***
- ***4 interfaces USARTs et 2 UARTs***
- ***3 interfaces SPIs***
- ***2 Interfaces CAN***
- ***1 interface SDIO***

Connectivité Avancée

- ***USB 2.0 full-speed (12 Mbps) pouvant être contrôleur du bus***
- ***USB 2.0 high-speed (480 Mbps) pouvant être contrôleur de bus avec DMA dédié***
- ***10MHz/100MHz Ethernet avec DMA dédié***

Interface parallèle pour caméra 8 à 14 bits

Générateur de nombres aléatoires vrai

Unité de Calcul de CRC

Identifiant unique de 96 bits

RTC avec précision inférieure à la seconde et calendrier matériel.

2.1 Discussion

Beaucoup de concepts et de termes très spécialisés sont présents dans la description générale du STM32F407. Voici quelques explications additionnelles reliées aux caractéristiques ci-dessus.

2.1.1 Cœur ARM ® Cortex ™-M4

Le coeur

Deux approches existent lorsqu'on veut concevoir un microcontrôleur :

1) Concevoir le microprocesseur, la mémoire et les périphériques du microcontrôleur
2) Obtenir une licence pour intégrer un cœur existant dans le microcontrôleur (le microprocesseur) et y ajouter de la mémoire et des périphériques.

Certaines compagnies comme ARM, MIPS, ARC vendent de la propriété intellectuelle (des designs de cœurs) plutôt que des microprocesseurs.

Le STM32F407 est un exemple de cette idée : le cœur a été conçu par ARM, mais le microcontrôleur appartient à ST Micro.

L'accélérateur d'accès à la FLASH (ART)

Le temps de lecture de la mémoire FLASH est habituellement entre 10ns et 20ns. Pour cette raison, 50MHz est une vitesse limitant souvent les microprocesseurs plus vieux ou très simples : il est impossible de lire plus qu'une instruction par coup d'horloge (en FLASH) lorsque l'horloge du cœur dépasse cette vitesse.

Pour contourner cette difficulté, il existe plusieurs solutions :

1. La solution la plus simple consiste à ajouter des Wait State dans le pipeline du microprocesseur. Le cœur attend après la FLASH. Évidemment, cette solution ralentit l'exécution des programmes…

2. Une autre solution commune est d'exécuter le code critique à partir de la SRAM, plus rapide que la FLASH en lecture (environ 5 fois plus rapide). Cependant, cela requiert deux copies du code à exécuter (une en mémoire non-volatile et l'autre en RAM). Cela requiert aussi du support matériel : le micro doit pouvoir exécuter du code en RAM. Enfin, lorsque le micro a un pipeline d'instruction et qu'une instruction LOAD ou STORE fait accès à la mémoire RAM, cela peut occasionner des stalls de pipeline : la plupart des systèmes ne permettent pas de lire une instruction en RAM tout en lisant ou écrivant une donnée en RAM.
    a. Il est possible d'exécuter du code à partir de la SRAM sans stall de pipeline si on ajoute du matériel… Par exemple, le XMC4400 implémente une mémoire RAM additionnelle pouvant être connectée au bus D-Code ou au bus I-Code!
3. On peut aussi ajouter une cache sur la mémoire FLASH et pré charger les instructions à exécuter. Il s'agit de la solution retenue pour le STM34F3207 : un bus 128 bits relie la FLASH à la cache d'instruction et permet de lire des instructions 16 bits ou 32 bits dans une cache assez rapidement pour fournir des instructions à 168MHz sans stall de pipeline (voir la Figure 4 de DM00031020.pdf).
    a. La plupart des caches d'instructions fonctionnent de pair avec des unités de prédiction des branchements. Lors d'énoncé conditionnel ou branchement, il faut prédire la séquence d'exécution des instructions afin de pré charger les instructions correctement.

Les DMIPs

Plusieurs facteurs déterminent la performance d'un microprocesseur : sa vitesse d'horloge, le nombre d'instructions maximum lancées simultanément, le nombre de coups d'horloge moyen par instruction, la nature des instructions exécutées, et plus.

La mesure de performance des microprocesseurs se fait habituellement avec des benchmarks (bancs d'essais) : il s'agit d'un ensemble de programmes exécutés par un microprocesseur. Le temps pris par un microprocesseur pour exécuter chaque programme est comparé au temps pris par les autres microprocesseurs et on obtient la performance relative de chaque processeur.

Les MIPS sont aussi une mesure de la performance d'un microprocesseur : il s'agit du Million d'Instructions Par Seconde. Ce n'est pas un indicateur parfait parce que les instructions sont souvent différentes d'un microprocesseur à l'autre, mais le nombre de MIPS est un meilleur estimé que la fréquence d'horloge…

Les DMIPs sont des MIPS évalués sur un benchmark, celui de Dhrystone (1984, Reinhold P. Weicker). Il s'agit du nombre de millions « d'instructions Dhrystone» exécutées par seconde avec des programmes fournis par le benchmark.

Instructions DSP

En traitement de signal, il est fréquent de faire des multiplications et des additions sur des fractions (faire une FFT par exemple!). Plusieurs instructions spéciales comme

l'instruction MAC (Multiply-Add-Cumulate) permettent d'accélérer le Digital Signal Processing.

2.1.2 Mémoire intégrée au circuit (On-Chip memory)

Afin de réduire les coûts du circuit intégré, les manufacturiers offrent plusieurs versions du même circuit : chaque version a une quantité différente de mémoire FLASH ou mémoire RAM.

En plus de la mémoire FLASH et de la mémoire RAM mentionnées sur la première page de la datasheet, le microcontrôleur contient aussi de la ROM interne exécutée au démarrage du microprocesseur (Bootloader). Cette ROM contient des instructions permettant de programmer la FLASH. Elle contient aussi certaines fonctions pour accéder aux périphériques et vérifier l'intégrité de la mémoire.

2.1.3 Contrôleur de mémoire statique flexible

L'accès à la mémoire d'instruction ou de données doit se faire avec du matériel spécialisé et des séquences spécifiques de signaux entre le microprocesseur et la mémoire. Comme il existe plusieurs types de mémoire (voir les prochains cours), il existe plusieurs séquences différentes de signaux pour accéder à celles-ci…

2.1.4 Interface Parallèle de LCD

Pour contrôler un écran, il faut soit un programme complexe, soit du matériel qui gère l'affichage et simplifie beaucoup la programmation.

Il y a deux modes de contrôle principaux de l'affichage utilisés de nos jours : le mode Intel 8080 et Motorola 6800, tous deux issus des années 1980.

2.1.5 Contrôle d'horloge de reset et d'alimentation

Le STM32F407 supporte une alimentation allant de 1.8V à 3.6V. Il s'agit de l'alimentation des périphériques et de la mémoire. Par contre, le cœur a une alimentation plus basse.

Un régulateur interne (inclus dans le boîtier) convertit la tension externe en 1.2Vdc. Ce régulateur alimente la cœur du microprocesseur qui peut également être alimenté par une source DC externe.

Par ailleurs, le microcontrôleur intègre un circuit supervisant la mise sous tension et la coupure d'alimentation (POR = Power-ON Reset, PDR = Power Down Reset). Ce circuit s'assure que le cœur est toujours convenablement alimenté lors d'exécution d'instructions. Le circuit de monitoring de voltage offre aussi des services de détection de faute d'alimentation (PVD = Programmable Voltage Detector) et de reset lors de chutes brèves de tension (BOR = Brown Out Reset).

La plupart des microprocesseurs requièrent une horloge externe pour fonctionner. Une horloge interne, intégrée au microcontrôleur, est parfois utilisée pour réduire le nombre de composantes requis afin de faire fonctionner le système microprocesseur. Cela assure aussi le démarrage du microprocesseur (les horloges externes peuvent ne pas fonctionner).

La précision de l'horloge 16MHz du RC intégré dans le microcontrôleur est de 1%, ce qui est très bien pour une horloge interne, mais médiocre comparé à une horloge externe avec cristal. Si on se sert de cette horloge pour compter le temps, elle générerait 1 seconde d'erreur à toutes les deux minutes!

Par ailleurs, le microcontrôleur accepte une horloge indépendante et plus basse fréquence pour le Real-Time Clock : il s'agit d'un circuit qui mesure le temps avec précision, souvent alimenté par une batterie.

2.1.6 Basse Puissance

Le microcontrôleur supporte trois modes d'opérations spéciaux afin de réduire la consommation d'énergie du micro :

- Sommeil : Le cœur est arrêté (sans horloge), le reste du microcontrôleur fonctionne normalement. Le micro sort du sommeil lors d'une interruption.
- Arrêt : Toutes les horloges sont arrêtées (horloge du cœur, des bus, des préiphériques…). Le micro redécolle sur activation d'une broche externe préconfigurée.
- Standby : Le régulateur 1.2V du cœur et des mémoires est coupé. Toutes les horloges ne fonctionnent plus. Le micro redécolle sur un signal du RTC.
    - Une partie de la mémoire RAM (backup RAM) et une copie des registres du microprocesseur conservent leur valeur en mode Standby.

Le microprocesseur quitte les modes basse-puissance lors de reset.

Il est aussi possible de réduire la consommation énergétique du microcontrôleur en diminuant les fréquences des horloges ou en coupant l'horloge des périphériques inutilisés.

2.1.7 ADCs et DACs

Un ADC convertit des tensions analogiques en valeurs digitales. Un ADC 12-bits avec une référence à 3.3V convertira une tension entre 0 et 3.3V en une valeur digitale, proportionnelle à la tension, entre 0 et 4095 ($2^{12} = 4096$).

Selon les critères actuels, un ADC 12-bits avec une fréquence d'échantillonnage de 2.4MHz maximum (MSPS = Mega Samples Per Second) est un bon ADC, sans être exceptionnel. La possibilité d'entrelacer les ADCs pour obtenir une fréquence de 7.2MHz est très intéressante par contre.

Un DAC, Digital to Analog Converter, convertit des signaux digitaux en signaux analogiques. Un DAC 12 bits ayant une tension de référence de 3.3V peut produire des échelons de tensions aussi petits que 3.3V/4096 = 0.8mV, en théorie.

2.1.8 DMA

Le DMA du STM32F407 lui permet de transférer des données d'une interface (i.e. ADC, SPI, USB, UART, etc.) vers la mémoire ou vice-versa, sans intervention du microprocesseur.

Le DMA supporte 16 canaux (il est possible de programmer 16 transferts) et contient des files (FIFO = First In First Out) pour faciliter les transferts de données. Le contrôleur de DMA permet aussi de gérer des données en rafales.

2.1.9 Minuteries et RTC

Les timers servent à mesurer ou calculer des intervalles de temps dans un système microprocesseur. Le nombre de bits d'un timer indique la valeur maximum que peut attendre ce timer avant de faire un overflow (ce qui déclenche habituellement une interruption).

Le STM32F407 intègre jusqu'à 16 timers qui lui sont propres (12 timers 16-bits, 2 timers 32 bits qu'il est possible de séparer en deux) et le systick timer propre au cœur ARM-Cortex M4.

Ces timers peuvent servir de minuteries, de PWM, de compteur de pulse, de modulateur/démodulateur, de watchdog ou pour contrôleur les phases d'un moteur en quadrature.

> *Un PWM (Pulse Width Modulation) est un circuit qui génère des pulses d'une durée prédéterminée à une fréquence prédétermine. Par exemple, la sortie d'un PWM générant des pulses de 1 ms seconde à une fréquence de 500Hz sera une onde carrée dont la fréquence est 500Hz.*
>
> *Un chien de garde (watchdog) est un circuit qui cause un reset du microprocesseur lorsqu'il n'est pas nourri. Lorsque le watchdog est activé, il faut le nourrir constamment sous peine de subir un reset.*

Par ailleurs, le STM32F407 possède un RTC, c'est-à-dire un circuit intégré qui compte le temps en secondes, minutes, heures, jours, semaines… Le RTC a sa propre horloge très précise et une alimentation indépendante pour continuer de compter le temps avec une batterie quand l'alimentation principale est éteinte.

2.1.10 Environnement de l'architecture

Supporte le déverminage par JTAG et ARM Serial Wire Debug (SWD)

Les interfaces JTAG et SWD sont des interfaces séries qui communiquent directement avec le cœur ARM ou la mémoire du microcontrôleur et permettent de déverminer les applications du microprocesseur en temps réel.

Nombre de broches et boîtier du circuit intégré (package)

Le STM32F407 peut avoir 64 pins, 90 pins, 100 pins, 144 pins ou 176 pins. Il s'agit souvent du même die, mais dans différents boîtiers de différentes tailles. Les formats plus petits coûtent moins cher et sont plus facile à souder. Cependant, les formats plus gros ont plus d'entrées/sorties et peuvent contrôler davantage de périphériques.

Le STM32F0407 est disponible en format LQFP, UFBGA et WLCSP. Ce sont différents agencements des broches sur la surface du boitier. Dans un format LQFP, les broches sortent de chaque côté du boitier. Pour UFBGA et WLCSP, des billes sous le boîtier relier le circuit au PCB. Le format WLCSP se distingue du format UFBGA par la taille du boîtier : il est beaucoup plus petit, presque de la dimension du die. Ces différents formats sont illustrés sur la première page de la datasheet ainsi que plus loin.

Industriel. Température de -40ºC à 85ºC ou 105ºC
Beaucoup de circuits intégrés opèrent de 0 ºC à 70 ºC. Pour les usages industriels cependant, le circuit intégré doit fonctionner de -40 ºC à +85ºC. Il faut un environnement particulier pour justifier des températures d'opération jusqu'à 105 ºC.

2.1.11 Broches d'entrées/sorties

Un GPIO est une entrée/sortie digitale tout usage. Il s'agit d'une broche configurée en entrée ou en sortie : l'entrée permet de lire un "0" ou un "1" tandis que la sortie permet d'écrire un "0" ou un "1".

2.1.12 Interfaces de communication

Les interfaces de communication série du STM32F407 sont communes et plusieurs seront présentées en détail lors du cours sur le sujet:

- Le SPI et l'I2C sont des interfaces séries assez haute vitesse (1MHz à 10MHz) habituellement utilisées entre deux circuits intégrés sur de courtes distances (le même PCB).
- Le UART est la plus vieille interface série. Il s'agit du RS232 avec des niveaux de tension TTL ou CMOS. Les UARTs sont souvent accompagnés de convertisseurs de niveaux de tension pour obtenir des bus RS232, RS422 ou RS485. Un USART est un UART avec plus de fonctionnalités.
- Le CAN est aussi une interface série très robuste principalement utilisée dans les véhicules ou les milieux industriels.
- L'interface SDIO sert à lire des cartes de mémoire SD.

2.1.13 Connectivité Avancée

Le microcontrôleur contient des circuits électroniques pour communiquer par USB ou Ethernet.

Les transferts à très haute-vitesse de ces bus ne peuvent pas être gérés par le microprocesseur qui est plus lent que 480Mbps ou incapable de faire un transfert à 100MHz : des canaux de DMA spéciaux et des bus spéciaux permettent de transférer les données de ces bus vers la mémoire ou vice-versa.

2.1.14 Interface parallèle pour caméra

Du matériel spécialisé à l'intérieur du microcontrôleur permet de traiter des signaux vidéos: synchronisation des trames, compression/décompression, modes continus et sanpshots, interprétation des formats d'image…

2.1.15 Générateur de nombres aléatoires vrai

Générer un nombre purement aléatoire est impossible dans un monde entièrement digital. Pour qu'un nombre soit aléatoire, il faut que les probabilités d'obtenir toutes les valeurs possibles du nombre soient égales. Il faut aussi que la séquence de nombres obtenus soit aléatoire et qu'on ne puisse pas prédire le ou les prochains nombres en fonction du passé.

Habituellement, en électronique, les nombres aléatoires sont pseudos-aléatoires : une graine aléatoire (seed) est utilisée pour calculer le prochain nombre de la séquence pseudo-aléatoire selon une formule mathématique prédéterminée. La formule mathématique est déterminée de telle sorte que la séquence de nombres aléatoires ne se répète qu'après un très grand nombre de valeurs et pour que les nombres sortant varient beaucoup.

On obtient aussi des nombres presque aléatoires en prenant des valeurs d'une horloge très haute-fréquence à un moment aléatoire. Cependant, il est possible de déterminer la prochaine valeur obtenue par cette méthode en fonction du passé.

Le générateur de nombres aléatoires du STM32F407 est basé sur une entrée analogique : le générateur de nombre lit du bruit…

2.1.16 Unité de Calcul de CRC

Un CRC, Cyclic Redondancy Check, est une valeur calculée sur un bloc de donnée. On prend chaque byte du bloc de données et on effectue un calcul mathématique:
Nouveau CRC = FonctionMathématique(Vieux CRC, Byte additionnel).

Habituellement, un CRC est calculé deux fois : une fois lorsqu'on écrit ou transmet le bloc de donné et une autre fois lorsqu'on lit ou reçoit le bloc de donnée. Si le CRC calculé la première fois (écrit ou transmis) n'est pas le même que le CRC calculé la seconde fois (à partir du bloc de données lu ou reçu), le bloc de donnée est corrompu.

L’unité de calcul de CRC permet de calculer automatiquement le CRC de certains blocs de données afin de sauver du temps de processeur lorsque le calcul doit être effcetuer rapidement.

2.1.17 Identifiant unique de 96 bits

Dans beaucoup de systèmes, les appareils faisant partie du système doivent pouvoir être identifiés. Par exemple, une MAC est nécessaire dans les réseaux Ethernet… Inclure un identifiant unique à l’intérieur du microcontrôleur simplifie la production d’appareils utilisant ce microcontrôleur.

3 Design de circuit avec microprocesseur

Lorsqu'on veut concevoir un appareil électronique, il faut d'abord choisir le ou les microcontrôleur(s) qui sera ou seront utilisé(s). Ensuite, il faut élaborer les circuits électroniques entourant le micro choisi. Peu ce dernier, il vous faudra toujours déterminer : les sources d'alimentation de votre appareil, les circuits d'horloge accompagnant le micro, les mémoires externes utilisées, les périphériques externes requis, l'électronique entourant la mise sous tension du circuit (ou un reset matériel) et la méthode utilisée pour mettre en mémoire les instructions exécutées par le micro. Habituellement, le déverminage du code fait également partie intégrante du design du circuit.

3.1 Le choix du microprocesseur ou du microcontrôleur

Lorsqu'on veut concevoir un système embarqué, une des premières questions qui apparaît est : quel micro dois-je choisir? Il existe une multitude de manufacturiers et une multitude de microprocesseurs pour chaque manufacturier. Il existe aussi plusieurs architectures de système microprocesseur. Choisir un microcontrôleur ou un microprocesseur est parfois difficile.

3.1.1 Déterminer ses besoins

Habituellement, la première étape de choix de micro est de déterminer ses besoins. Cette étape est cruciale et elle n'est pas toujours facile à franchir. Plusieurs questions doivent se poser pour choisir le microcontrôleur ou le microprocesseur. En voici une liste non-exhaustive :

- Le système supportera-t-il une application spécifique ou doit-il être générique pour plusieurs applications?
- Quelle puissance de calcul ai-je besoin?
    - Dois-je supporter un système d'exploitation?
    - Dois-je parfois exécuter plusieurs tâches simultanément?
- Quelle quantité de mémoire pour les données (RAM) est requise ?
- Quelle quantité de mémoire pour le code (ROM) est requise ?
    - Est-il désirable de pouvoir reprogrammer un
- La mémoire non-volatile contiendra-t-elle des données ?
- Quelles sont les entrées/sorties et interfaces de communication dont j'ai besoin?
- Quelle est la puissance disponible?
- Quelle est l'espace physique disponible pour mettre le système microprocesseur?
- Quelles sont les contraintes reliées à l'environnement du microprocesseur (température d'opération, résistance aux bruits, etc.)?
- Puis-je avoir des outils afin de développer efficacement le logiciel et le matériel pour ce microprocesseur?
- Quel est le prix désiré en fonction des quantités désirées?

3.1.2 Choisir le microcontrôleur ou le microprocesseur

Souvent, plusieurs systèmes microprocesseurs rencontrent nos besoins. Il est typique de rencontrer plusieurs solutions qui s'équivalent. Dans ce cas, il convient d'utiliser d'autres critères de sélection du microprocesseur plus subjectifs : ai-je déjà utilisé le microprocesseur ou un micro de la même famille? Existe-t-il des exemples de code pour des applications similaires ou le produit est-il utilisé dans des applications similaires? Est-ce que j'utilise d'autres circuits intégrés de ce manufacturier ? Existe-t-il des kits d'évaluation qui me permettront de réduire les coûts et le temps de développement d'éventuels prototypes ? …

Lorsqu'un microprocesseur a été choisi, il faut insérer ce microprocesseur à l'intérieur d'un circuit numérique. Plusieurs points sont à considérer dans le design du circuit autour du microprocesseur. Les principaux points à déterminer sont : l'alimentation du circuit, l'horloge, les bus d'accès à la mémoire, les interfaces avec les entrées/sortie, et comment programmer la mémoire d'instruction …

3.2 Alimentation

La plupart des microprocesseurs modernes ont plusieurs broches d'alimentation en courant continu.

Les principales broches d'alimentation servent à alimenter le cœur du microprocesseur lui-même. En opération normale, le cœur consomme une puissance proportionnelle à sa fréquence d'opération (*f*) et au carré de son voltage d'alimentation (*V*):

$$P_{total} = P_{dynamique} + P_{statique} \approx K_1 * f_{coeur} * V^2 + K_2 V$$

Dans cette équation, K_1 et K_2 sont des constantes essentiellement déterminées par les caractéristiques et le nombre de transistors du microprocesseur. Par ailleurs, f_{coeur} est la fréquence d'opération du cœur du microcontrôleur.

Le microprocesseur requiert de l'énergie même lorsque l'horloge ne fonctionne pas, mais cette énergie (K_2V*t) est très petite par rapport à l'énergie consommée lorsqu'il y a une horloge de plusieurs MHz.

La source d'alimentation DC pour le cœur devrait être suffisamment puissante pour alimenter le cœur à sa fréquence d'opération prévue et pour pouvoir fournir davantage afin de tenir compte d'effets thermiques ou d'éventuels ajouts.

Habituellement, les microprocesseurs modernes auront également des broches d'alimentation supplémentaires pour les entrées/sorties. En effet, les entrées/sorties sont très souvent 3.3V ou 5V pour des raisons de compatibilité avec le matériel électronique existant alors que le cœur du micro a une tension d'opération beaucoup plus faible (pour diminuer la consommation de puissance; exemple : 1.3V). Par ailleurs, certaines entrées/sorties peuvent fournir/tirer des courants assez larges et il apparaît comme une bonne pratique de séparer les alimentations dans ce cas.

Lorsque beaucoup de courant circule par une broche, cela crée de la chaleur localement (il y a des pertes par résistance autour de la broche). Séparer les alimentations et avoir plusieurs broches répartit la chaleur. Cela répartit aussi le bruit : le bruit autour d'une broche d'alimentation peut être isolé du bruit autour d'une autre broche. Enfin, éviter des courants larges circulant par des broches uniques d'alimentation permet de mieux dimensionner les broches et d'éviter des tensions parasites introduites par les résistances des fils et connecteurs du circuit.

Ensuite, les microprocesseurs modernes ont souvent des broches d'alimentations indépendantes pour les interfaces ou entrées/sorties qui requièrent une alimentation particulièrement précise et/ou stable. Par exemple, la tension de référence d'un ADC (Analog to Digital Converter) doit être très stable pour éviter de fausser la mesure en raison de fluctuations de l'alimentation.

Finalement, il y aura potentiellement des broches d'alimentations supplémentaires pour les systèmes embarqués utilisant des batteries. Par exemple, il peut y avoir une broche d'alimentation servant à alimenter le RTC (Real Time Clock) avec une pile lorsque le système microprocesseur est éteint.

3.3 Horloge

Le concepteur du circuit numérique doit choisir une horloge pour son microprocesseur. Les choix sont les suivants :

- Horloge interne ou externe : les horloges internes sont peu précises. En contrepartie, il faut des composantes additionnelles pour les horloges externes (très peu) et le temps de démarrage ou le démarrage lui-même des horloges externes peut parfois être un problème.
- Fréquence d'horloge : Plus la fréquence d'horloge est grande, plus le microprocesseur sera rapide. Souvent, la fréquence d'horloge du microprocesseur déterminera aussi la fréquence d'opération de plusieurs bus du système et la fréquence d'opération de plusieurs périphériques. Plus elle est élevée, plus le système sera rapide. Cependant, la consommation de puissance est directement proportionnelle à la fréquence et certaines composantes ont une fréquence d'opération maximale.

3.4 Mémoire et accès à la mémoire

Tout système microprocesseur requiert une mémoire pour contenir les instructions à exécuter et les données traitées. Pour relier le microprocesseur à la mémoire, il faut : déterminer la vitesse de communication sur le bus de la mémoire; déterminer les adresses réservées à la mémoire et concevoir un circuit de décodage d'adresse; puis relier les bus de données et d'adresse en considérant un éventuel multiplexage temporel des broches d'adresses et de données.

Pour diminuer le nombre de broches requises sur le microprocesseur, il arrive fréquemment que certaines broches –souvent baptisées AD- servent d'adresse dans un premier temps, puis de donnée dans un second temps.

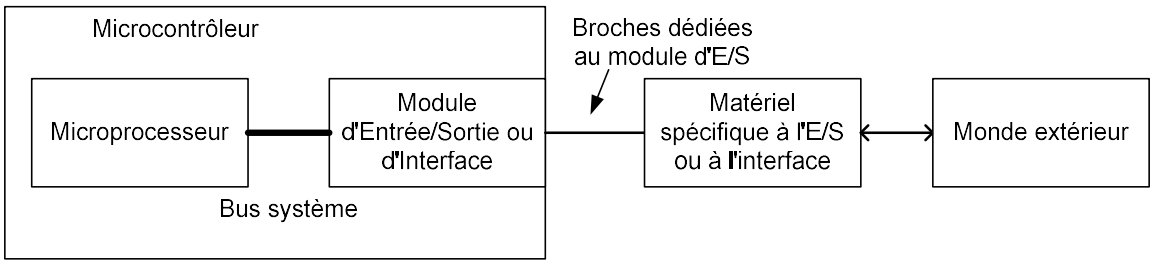
Dans les microcontrôleurs modernes, de plus en plus de mémoire est intégrée dans le même circuit que le microprocesseur de telle sorte que les petits microcontrôleurs n'ont souvent pas de bus parallèle pour la mémoire externe. Avoir beaucoup de mémoire dans le microcontrôleur permet d'éviter d'utiliser des broches, d'économiser des coûts et d'économiser de l'espace pour la mémoire et pour son filage.

En contrepartie, le microcontrôleur perd de la versatilité : il devient difficile plus difficile de changer la quantité de mémoire dans le système. Par ailleurs, il se peut que l'on paye pour de la mémoire superflue... Ces désavantages ne sont toutefois pas majeurs : il reste possible d'ajouter des mémoires séries, le coût de la mémoire est relativement faible et, si le microcontrôleur a bien été choisi, des microprocesseurs de la même famille que le microprocesseur choisi, avec des broches identiques, auront plus de mémoire.

Combien faut-il de broches, au minimum, pour gérer les adresses et les données d'une mémoire externe de 256K * 16 bits?

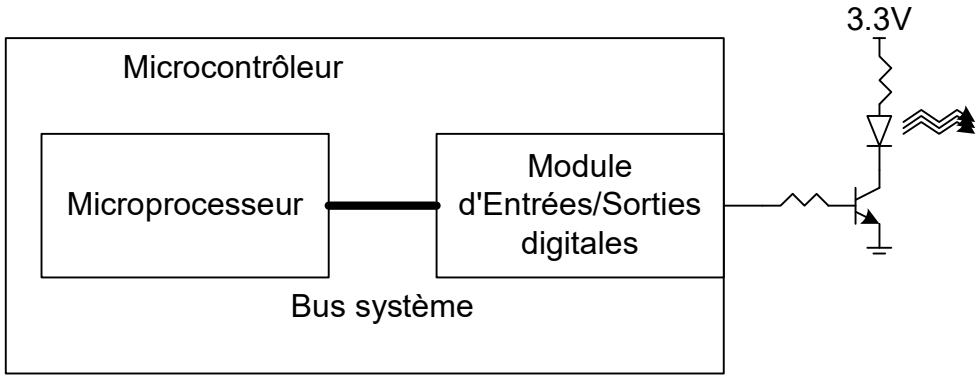
3.5 Entrées/Sorties

Brancher un microcontrôleur à ses entrées/sorties est habituellement très simple : des modules d'E/S existent pour la plupart des périphériques et il ne manque que quelques composantes afin de faire le lien entre le microprocesseur et le monde extérieur.

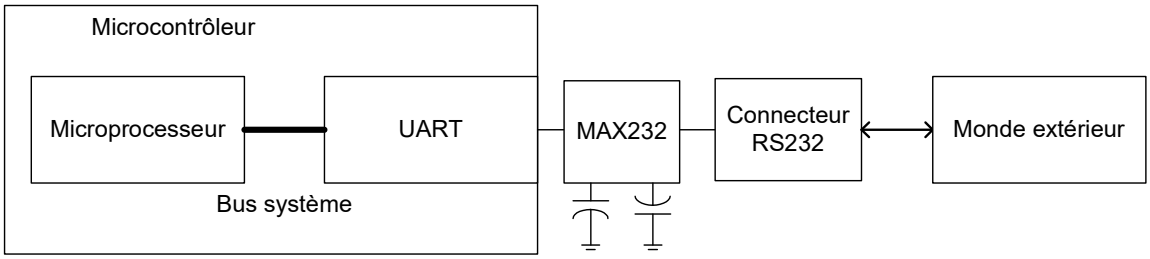


Voici deux exemples illustrant ce propos : une LED et un port série RS232.

Tous les microcontrôleurs ont des entrées-sorties tout usage (GPIO) contrôlée par le module d'entrées/sorties digitales. Certaines instructions visant les registres du module d'E/S digitales permettront d'imposer 0V ou 3.3V/5V sur une broche du microcontrôleur. Cependant, la broche ne pourra probablement pas fournir assez de courant pour allumer la LED : il faudra éventuellement ajouter un transistor pour que le courant allumant la LED ne provienne pas de la broche…



Le port série quant à lui, sera contrôlé par le module d'E/S UART dont le principal rôle est de mettre en série des octets parallèle et vice-versa. Les bits sortant du module UART sont conformes à la norme RS232, mais les niveaux de tension sont incorrects (le microcontrôleur fonctionne avec du 3.3V/5V et la norme prescrit ± 15V). C'est pourquoi il faut ajouter un MAX232…



3.6 Circuit de reset

Le concepteur d'un circuit avec microprocesseur doit toujours prendre soin de la broche de reset !

Il convient de généralement protéger cette broche contre le bruit pour éviter des redémarrages indésirés du microprocesseur. La plupart des microcontrôleurs modernes intègrent des protections sur la broche reset, mais il est d'usage général de mettre un circuit RC sur la broche afin de réduire du bruit.

La broche reset des microprocesseurs est parfois reliée au circuit de watchdog externe ou au circuit de programmation de la mémoire.

3.7 Programmation de la mémoire d'instruction

Que la mémoire d'instruction soit interne (dans le microcontrôleur) ou externe, il faut prévoir une méthode pour la programmer.

Les mémoires d'instructions externes peuvent être programmées en dehors du système: avant d'être soudée ou si elle est montée sur socket, la mémoire peut être effacée et écrite avec le code voulu.

Les mémoires d'instructions internes (et parfois externes aussi) sont habituellement programmées à l'aide de programmes également en mémoire non-volatile… Par exemple, un programme dans une mémoire ROM (EEPROM) s'exécutera au démarrage si certaines conditions électriques sont rencontrées sur des broches du microprocesseur. Ce programme gèrera le port série et interprètera des commandes provenant de l'extérieur et écrivant la mémoire FLASH.

Dans les plus vieux systèmes microprocesseurs, la mémoire contenant les instructions était retirée du système avant d'être reprogrammée (exemples : cartes performées ou SMI avec UV-PROM). Cependant, il est impossible de retirer la FLASH interne d'un microcontrôleur et la méthode est très laborieuse… Aussi, les plus vieux microcontrôleurs se programmaient presque tous par le UART exclusivement. Les microcontrôleurs actuels se programment encore par le UART, mais d'autres interfaces plus rapides permettent aussi souvent de le faire (les mémoires d'instruction sont plus grandes et le UART devient lent !). On retrouvera communément une interface additionnelle comme l'I2C ou le CAN.

Enfin, les mémoires peuvent aussi être reprogrammée dans un circuit aux composantes déjà soudées autrement que par une séquence d'instructions exécutée par le microprocesseur : dans certains circuits et pour certains cœurs, il est possible prendre le contrôle du bus système et d'écrire directement la mémoire. Par exemple, le JTAG permet d'obtenir ce résultat.

3.8 Méthode de déverminage du code

Avec des systèmes microprocesseurs de plus en plus intégrés, les circuits électroniques entourant le microcontrôleur sont de moins en moins nombreux et le code peut devenir de plus en plus complexe. De ce fait, il est souvent primordial de prévoir du support au debug dans les prototypes électroniques que vous concevrez. La principale interface de debug utilisée avec les microcontrôleurs modernes est le JTAG (Join Test Action Group) qui sera présenté ultérieurement. Le SWD (Serial Wire Debug) est aussi utilisé énormément, mais il ne sera pas présenté car il est similaire au JTAG.

4 Annexe A: Design de circuit avec microprocesseur et STM32F407

Les questions qui suivent vous permettront de mieux connaître le STM32F407, utiliser dans les laboratoires du cours. Elles regroupent les questions devant être posées lorsqu'on conçoit un circuit avec un microcontrôleur particulier.

Le lecteur est invité à ouvrir la fiche technique (datasheet) du microcontrôleur afin d'y chercher les réponses qui se retrouvent également à la page suivante.

4.1.1 Questions reliées au STM32F407

Q1) Quelles sont les broches d'alimentations du STM32F407? Quelle est la consommation maximale de courant sur ces broches ?

Q2) Quelles sont les fréquences d'horloge possible pour le STM32F407? Pour quelles raisons brancherions-nous plusieurs horloges externes sur le microcontrôleur?

Q3) Le STM32F407 peut-il adresser de la mémoire externe? Si oui, quelle quantité peut-elle être adressée et par quelles broches?

Q4) Le STM32F407 a-t-il un reset actif HIGH ou actif LOW? Pourquoi? Existe-t-il un filtre interne sur le circuit de reset?

Q5) Comment peut-on programmer la FLASH interne du STM32F407?