Pour comprendre : comparatif Arduino+shields / Pyduino + mini-PC

Ateliers Pyduino

par X. HINAULT
www.mon-club-elec.fr

Dans ce document, afin de vous permettre de prendre conscience de l'intérêt potentiel de l'utilisation de la librairie Pyduino avec un mini-PC, vous trouverez une tentative de comparatif des possibilités d'un système Arduino + shield et d'un système Pyduino + mini-PC.

1. Synthèse comparative des possibilités Arduino+shields versus Pyduino+mini-PC

Fonctions de gestion E/S		
Sorties numériques ON/OFF	20 broches E/S potentielles sur la carte UNO – en 5V	18 E/S sur le pcDuino, 8 E/S sur le RPi – en 3.3V... compatible avec niveaux logiques dispositifs 5V..
Entrées numériques ON/OFF	rappel au plus interne en entrée sur toutes les broches E/S	rappel au plus interne en entrée sur toutes les broches E/S
Entrées analogiques	6 broches analogiques en résolution 10 bits en 5V	pcduino : 6 broches analogiques dont 4 en résolution 12bits en 3.3V RPi : 0 broches analogiques
Sorties « analogiques » : impuls. PWM	6 voies PWM disponibles / 520Hz	pcduino : 6 broches PWM / 520 Hz- freq ajustable RPi : 1 broche PWM
Fonctions de gestion dispositifs		
Comm Série USB/UART	Communication USB native via 2 broches E/S	Oui aussi bien par USB que UART matérielle
Comm série I2C	Oui, avec librairie Wire – Librairies dédiées nombreuses	Oui, à venir - Librairie dédiées peu nombreuses
Comm série SPI	Oui, avec librairie SPI - Librairies dédiées nombreuses	Oui, à venir - librairies dédiées peu nombreuses
Comm série spécialisées (1-wire, DHT...)	OUI, avec librairie dédié	Plutôt NON, mais possibilités existantes, variable selon carte miniPC. 1-wire et DHT ok sur RPi
Impulsion servos	12 broches utilisables pour impulsions servomoteur (librairie Servo)	OUI, via PWM avec cependant stabilité impulsion moindre, mais correcte – maxi 4 servos
Afficheurs LCD	OUI (librairie LiquidCrystal)	OUI, librairie LiquidCrystal implémentée dans Pyduino pour afficheur LCD standard
Contrôle de motorisation CC et pap	OUI, à l'aide interface adaptée	OUI à l'aide interface adaptée
Fonctions « système »		
Fichiers texte	Oui, avec shield SD complémentaire. Possibilités limitées	Oui, SANS SHIELD COMPLEMENTAIRE – gestion simple de fichiers volumineux, répertoires, etc...
Réseau	Oui, avec shield Ethernet complémentaire. Stabilité modérée. Reboot physique si plantage.	Oui, SANS SHIELD COMPLEMENTAIRE –
Multitâche	NON : un seul code exécuté à la fois !	OUI : plusieurs codes peuvent être exécutés simultanément en parallèle
Programmation distante via réseau	NON : pas de programmation distante possible	OUI : par accès au bureau distant
Prise de contrôle distante via réseau	NON	OUI : par accès au bureau distant, permettant de relancer un code « serveur » de façon distante !
Librairie Javascript locale	NON : nécessité serveur externe	OUI : hébergement de librairie javascript et de tout fichier utiles sur serveur Http local
Utilisation de dispositif USB standards	NON . Shields dédiés coûteux obligatoires pour les fonctions spécifiques (GPS, etc..)	OUI : notamment numpad, GPS, etc.. Grosses Economies possibles ++
Connexion sans fil	OUI avec shield complémentaire, parfois très cher (wifi)	OUI à l'aide d'une clé wifi USB standard (RPi) ou dédiée (pcDuino) et même intégrée (pcduino 2) !!
Fonctions « multimédia»		
Fichiers images	Support possible fichier image en format simplifié avec shield carte SD	OUI : Support natif de fichiers image aux formats standards
Fichiers sons	Support possible de format sons dégradés avec shield complémentaire	OUI : Support natif de fichiers sons aux formats standards
Fichiers Vidéo	A priori non	OUI : Support natif des fichiers vidéos aux formats standards
Capture / lecture audio	Plutôt NON ; Possible à minima à l'aide de shields dédiés chers	OUI : Capture sonore possible Lecture fichier sons et bruitages avec carte son USB
Capture / lecture voix	Plutôt NON : Possible à minima à l'aide de shields dédiés chers	OUI : Synthèse vocale native, Reconnaissance vocale en mode connecté
Capture / lecture image	Plutôt NON : Possible à minima à l'aide de shields dédiés chers	OUI : Capture d'image fixe avec possibilités de traitement
Capture / lecture vidéo	NON	OUI : Capture vidéo possible, lecture vidéo possible

Un ensemble Arduino + shields à fonctionnalités « système » + « multimédia » équivalentes coûterait 2 à 3 fois le prix d'un mini-PC tout en faisant moins bien au final !

2. *Synthèse comparative des fonctions « multimédia »*

 Non disponible !  Nécessite Shield !  Ready !

		
		
		
		
		
		

3. Les atouts clés d'un mini-PC

En plus de disposer de la plupart des fonctionnalités d'une carte Arduino, un mini-PC :

- est utilisable d'emblée en réseau donc de façon distante
- est contrôlable et programmable à distance, y compris par wifi (notamment le pcduino v2)
- ne nécessite aucun shield supplémentaire pour utiliser les fichiers texte, audio, vidéo
- ne nécessite aucun shield supplémentaire pour utiliser le réseau ethernet voire le wifi (une clé USB wifi est suffisante quand le wifi n'est pas déjà intégré!)
- ne nécessite aucun shield supplémentaire pour audio, la voix, une webcam, etc..

Et aussi :

- permet de mettre en place un serveur graphique sans avoir besoin de serveur externe !
- est contrôlable facilement par une tablette, smartphone, etc...

4. Général

Espace de développement	IDE Arduino basé sur Java = inutilisable pour ainsi dire sur un mini-PC	IDE Pyduino léger utilisable directement sur le mini-PC
Matériels utilisables	Shields spécifiques pour des fonctions dédiées précises	Matériel USB standard possible sous réserve de compatibilité et donc économies à la clé

5. Caractéristiques techniques générales de la base matérielle

Mémoire RAM disponible	Avec Arduino, on dispose de quelques quelques Ko ou kilo-octets : c'est largement suffisant pour les besoins simples, vite limité pour les chaînes de caractères notamment. Il faudra en tenir compte pour ne pas faire « déborder » la RAM.	Avec Pyduino par contre, on dispose potentiellement de la mémoire RAM disponible du mini-PC, c'est à dire plusieurs centaines de Mo ou Mega-octets !! Autant dire que vous pouvez considérer que vous disposez d'une place quasi illimitée de mémoire RAM pour stocker les variables de vos codes. Un souci de moins à gérer...

6. *Le langage*

		
Principe général d'utilisation ?	La carte Arduino est une petite carte électronique programmable qui dispose de broches d'entrées/sorties numériques, de broches analogiques de mesure, etc... La carte Arduino est programmée à l'aide d'un programme qui va être écrit dans un logiciel sur un poste fixe. Le programme va ensuite être compilé puis transféré par connexion USB vers la carte Arduino. La connexion entre le poste fixe et la carte Arduino se fait par USB et sert à programmer la carte Arduino.	Une carte mini-PC est une carte électronique qui est une sorte de « mini » carte mère de PC qui dispose de tous les éléments d'un PC classique (port USB, Ethernet, etc...) ainsi que d'entrées sorties numériques, de broches analogiques de mesure, etc... La carte mini-PC est programmée à l'aide d'un programme qui est écrit dans un logiciel directement exécuté sur le mini-PC. Le programme est directement exécuté sur le mini-PC. La connexion entre le poste fixe et le mini-PC se fait par réseau et sert à programmer/contrôler le mini-PC à partir d'un poste fixe.
Comment s'écrit le code ?	Un code Arduino s'écrit dans le logiciel Arduino qui intègre un éditeur à coloration syntaxique pour les fonctions Arduino.	Un code Pyduino s'écrit : • soit dans le logiciel Pyduino qui intègre un éditeur à coloration syntaxique pour les fonctions Pyduino • soit dans un éditeur « classique » à coloration syntaxique tel que Geany
Quel est le langage sous-jacent ?	Le langage Arduino est basé sur le langage C / C++ : c'est un langage compilé. Tout programme doit être compilé intégralement avant d'être exécuté : il ne doit comporter aucune erreur avant d'être exécuté. Il n'est pas possible de tester une instruction dans une console simplement.	La librairie Pyduino est basée sur le langage Python : c'est un langage interprété. Un programme s'exécute pas à pas et ne bloquera que lorsqu'une erreur sera rencontrée. Un intérêt majeur est la possibilité de tester une instruction à tout moment dans une console Python, sorte de Terminal interactif.

Apprentissage	Etant un langage compilé, Arduino ne permet pas de tester des instructions « à la volée » mais uniquement au sein d'un code.	En raison de son caractère « interprété », le langage Python offre un outil très intéressant pour l'apprentissage : la console Python (ou shell). Grâce à cet outil, il est possible de faire de tester des fonctions, des variables, etc... à la volée, en dehors de tout script, ligne à ligne. Ceci est très très utile en phase d'apprentissage et à tout moment pour tester un bout de code, découvrir une nouvelle fonction, etc... Envie de faire une mesure analogique ? Saisir simplement analogRead(A2) dans la console suffit !
Syntaxe	Plutôt « compliquée » notamment : > nécessité du ; de fin de ligne	Plutôt « simplifiée », hormis l'acquisition du principe de l'indentation par tabulation, notamment :

	> la gestion des correspondance des { } > l'obligation de préciser le type d'une variable ...	> pas de ; de fin de ligne > pas besoin de gérer les correspondance { } > pas besoin de préciser le type des variables : une réelle simplification pour le débutant !! ...
Gestion des types	Le langage Arduino est basé sur le langage C / C++ : le type de chaque variable doit être explicitement déclaré, ce qui peut être une source de complication inutile en phase d'apprentissage.	La librairie Pyduino est basée sur le langage Python : les types des variables est attribué automatiquement par l'interpréteur Python, simplifiant grandement l'utilisation des variables en phase d'apprentissage. L'accès au type explicite reste possible au besoin ainsi que les fonctions de changement de type.
Type et renvoi de fonction	Avec Arduino, le type de valeur renvoyée par une fonction doit être explicitement déclaré.	Avec Pyduino, le type de valeur renvoyée par une fonction n'a pas besoin d'être explicitement déclaré.
Type et paramètre de fonction	Avec Arduino, chaque valeur passée en paramètre à une fonction doit correspondre au type prédéfini. Les tableaux sont compliqués à utiliser en tant que paramètre de fonction	Avec Pyduino, le type des valeurs passées en paramètres n'est pas prédéfinis d'avance Les « tableaux » (objet list) peuvent être passés en paramètres de fonction sans aucun problème.
Les fonctions disponibles	Une 50 aines de fonction de bases Extensibles via des librairies dédiées	Les instructions Arduino de base Les instructions SD, Ethernet Les instructions d'horodatage Les instructions « multimédia » A venir : des instructions de librairies dédiées Déjà implémentées : LCD Servo Actuellement pas implémentée : I2C SPI
Portée didactique de l'apprentissage initial	Apprendre Arduino c'est apprendre le langage C qui est un langage plutôt « ancien » et présentant certains concepts complexes à maîtriser (les pointeurs, « rigidité » des tableaux, etc...). Bien que posant les bases pour l'apprentissage de d'autres langages tel que le Java, le Javascript, le C++, cet apprentissage nécessitera la prise en main d'outils différents pour des applications plus élaborées, notamment si l'on souhaite réaliser des interfaces graphiques.	Apprendre Pyduino, c'est poser les bases de l'apprentissage du langage Python, langage récent, qui est un « vrai » langage de programmation professionnel, puissant, aux possibilités avancées, simplifiant fortement certains éléments de programmation. C'est donc un apprentissage « rentable » qui conduira potentiellement très loin celui qui l'apprend, notamment pour la mise en place d'interfaces graphiques, l'utilisation de librairies scientifiques, le traitement d'image, etc... en utilisant le même langage ! Il est même possible d'intégrer les instructions Pyduino directement au sein d'un code d'une interface graphique ! Les librairies existantes en Python sont également très nombreuses et matures dans de très nombreux domaines.

7. Valeurs numériques

		
Taille maximale valeur entière	La taille maximale d'une valeur entière avec Arduino est le long qui peut stocker une valeur comprise entre de -2 147 483 648 à + 2 147 483 647	Pas de limite de la taille des valeurs entières
Précision en virgule flottante	La précision maximale supportée par Arduino pour les valeurs numériques à virgule est environ de l'ordre de 8 chiffres en tout avant/après la virgule.	Pour ainsi dire aucune limitation de la précision des nombres à virgule pour un usage usuel voire avancé ! Il est par exemple possible de poser des calculs utilisant des valeurs à 35 décimales derrière la virgule sans perte de précision !!

8. Fonctions mathématiques

		
Précision des calculs en virgule flottante	Avec Arduino, la précision en virgule flottante est limitée à 8 chiffres « en tout », en comptant les chiffres avant et après la virgule. Ceci n'est pas très gênant pour des calculs simples en virgule flottante... mais rend impossible des calculs astronomiques par exemple, en raison de la perte de précision.	Sur un mini-PC utilisé avec Pyduino, on dispose de la précision du langage Python pour les calculs en virgule flottante : il est dès lors possible d'utiliser des constantes physiques en puissance 10E-25 (25 chiffre derrière la virgule) ou plus, sans perte de précision.
Calcul « symbolique »	Non disponible avec Arduino.	Encore plus fort, le langage Python intègre une bibliothèque de calcul symbolique capable de gérer des équations 2x+3=0 par exemple sans réaliser d'arrondi dans le calcul !

9. *Chaînes de caractères*

		
Gestion de la mémoire RAM	Avec Arduino, les chaîne de caractères de messages sont stockées en mémoire RAM par défaut, ce qui ne pose pas de problème pour une dizaine de messages. Mais, dès que l'on va écrire beaucoup de messages, il faudra les écrire en mémoire FLASH (quelques dizaines de ko) pour ne pas faire déborder la mémoire RAM dont la taille est limitée (quelques ko). Ecrire un message en mémoire FLASH se fait depuis Arduino 1.0 par Serial.print(F("message"));	Comme pour Arduino, les chaînes de caractères de message sont stockées en mémoire RAM. Mais à la différence d'Arduino, avec Pyduino sur un mini-PC, **on n'a pour ainsi dire pas de limite de taille de la RAM disponible qui est généralement de plusieurs centaines de Mo disponibles !!!!** Ainsi, même si l'on écrit de nombreux messages texte dans un code, il n'y aura pas besoin de se soucier de la taille occupée en RAM !
Objet « chaîne de caractère »	Dans le langage Arduino, une chaîne de caractère pourra être gérée : • soit sous forme d'un tableau de caractères (type char) de taille fixe • soit sous forme d'un objet String qui offre plusieurs fonctions de manipulation simplifiée des chaînes de caractères Les objets String ne doivent cependant pas dépasser la taille de la RAM à la fois de part leur nombre et leur taille propre.	Avec Pyduino, l'objet str natif du langage Python est accessible directement dans le code. Le type char n'existe pas. L'objet str fournit de nombreuses fonctions de manipulation des chaînes de caractères, et est comparable à l'objet String de ce point de vue. Mais l'objet str dispose de possibilités beaucoup plus puissantes : • l'objet str permet des formatages avancés des chaînes • l'objet str peut être utilisé avec la notation dite « slice » qui permet une manipulation simplifiée et puissante des sous-chaînes au sein d'une chaîne • le langage Python dispose également d'une librairie qui permet l'analyse par « expressions régulières », outil très puissant pour l 'analyse de chaînes complexes. Avec Pyduino utilisée sur un mini-PC, il n'y a pour ainsi dire en pratique aucune limite de taille et de nombre des objets str() au sein d'un code, la RAM disponible étant « énorme ». L'objet str, enfin, supporte le format multiligne L'objet str multiligne supporte les caractères spéciaux au sein d'une chaîne sans besoin d'utiliser de caractère déchappement.

10. Comparatif de quelques caractéristiques matérielles

		
Tension des broches E/S	5V	3.3V
Freq max sur broches E/S entre 2 HIGH/LOW	plus de 50 000 / seconde	100µs par lecture/écriture soit 5000Hz environ max
Nombres de servomoteurs possibles	jusqu'à 12	jusqu'à 4 sur broches PWM
RAM programme	quelques ko	plusieurs Mo

11. Quand choisir un mini-PC + Pyduino ?

En bref :

pour tous les projets où les performances de bas niveau à « haute vitesse » et ou les contrôles de dispositifs ne sont pas déterminants (servomoteurs,..), préférer un miniPC

Pour tous les projets où la puissance de calcul, les fonctions « système » (fichier, réseau) et « multimédia » (voix, audio) sont prépondérantes ou souhaitées, choisir un miniPC