

Ecole Polytechnique de l'Université de Tours

64, Avenue Jean Portalis

37200 TOURS, FRANCE

Tél. +33(0)2 47 36 14 14

Fax. +33(0)2 47 36 14 22

[www.polytech.univ-tours.fr](http://www.polytech.univ-tours.fr/)

Compte-rendu du tp2

**Systèmes embarqués**

|  |  |
| --- | --- |
| Apprentis : | Enseignant : |
| Quentin Chalopin  [quentin.chalopin@etu.univ-tours.fr](mailto:quentin.chalopin@etu.univ-tours.fr) | Anthony Jouret  [anthony.jouret@univ-tours.fr](mailto:anthony.jouret@univ-tours.fr) |
| Julien Mouton  [julien.mouton@etu.univ-tours.fr](mailto:julien.mouton@etu.univ-tours.fr) |  |

Table des matières

**Tapez le titre du chapitre (niveau 1)1**

Tapez le titre du chapitre (niveau 2)2

Tapez le titre du chapitre (niveau 3)3

**Tapez le titre du chapitre (niveau 1)4**

Tapez le titre du chapitre (niveau 2)5

Tapez le titre du chapitre (niveau 3)6

# Introduction

Tout au long de ce compte-rendu, nous allons vous expliquer comment utiliser les scripts cgi sur la carte Armadeus. Nous allons également utiliser un port I2C pour communiquer avec un capteur d’humidité et de température. Les valeurs mesurées par ce capteur seront accessibles à travers une page internet grâce au serveur web également déployé sur la carte.

# Matériel

Pour réaliser tout cela, il nous faudra :

* Un ordinateur doté d’un linux ou d’une machine virtuelle linux : Outil pour pouvoir compiler et envoyer les données à la carte.
* 1 carte Armadeus APF27 (celle utilisée à Polytech’) : La carte où va être implanté le système d’exploitation.
* 1 convertisseur USB 🡪 Série : Très utile si nous n’avons pas de port série directement disponible sur notre PC.
* 1 câble série : La communication série nous permettra d’interagir avec le système.
* 1 câble Ethernet : Nécessaire pour la cross-compilation via l’ordinateur.
* 1 capteur TI HDC1000

# Mise en place

Pour mettre en œuvre tout cela, nous allons suivre le schéma suivant :

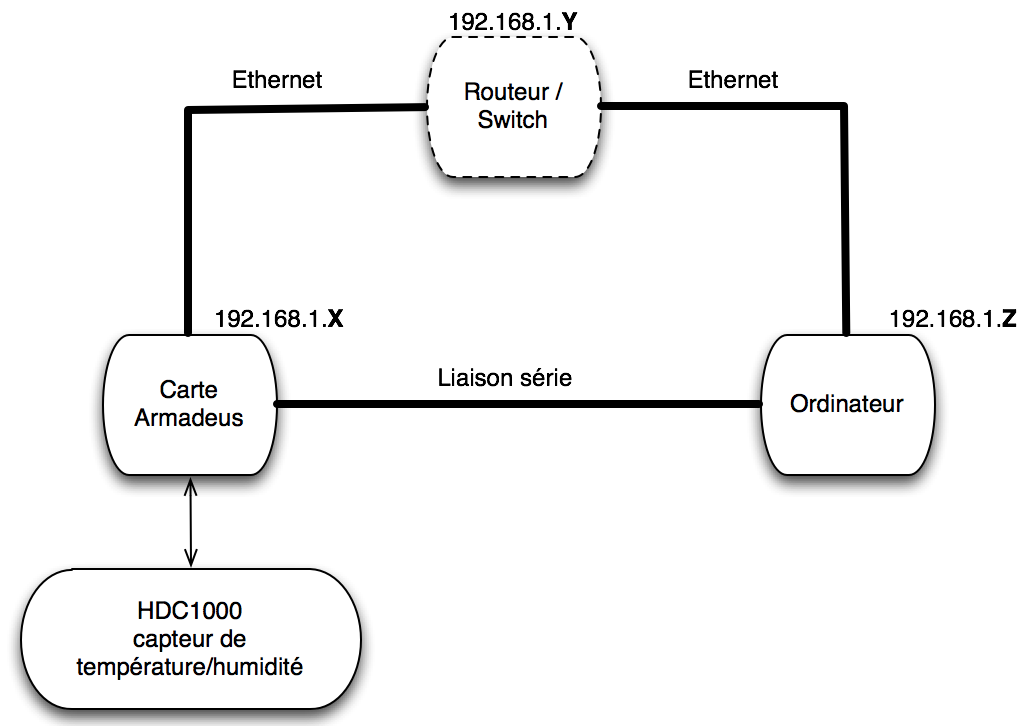
****

Figure 1 Architecture

**Remarque :** Le routeur (ou switch) n’est pas nécessaire. Une connexion point à point fera l’affaire.

Pour la **communication série**, aucun logiciel n’est recommandé. En ce qui nous concerne, nous allons utiliser **Hercules**.

**Remarque :** Si vous utilisez un adaptateur USB 🡪 Série, veillez à ce que vos pilotes soient bien à jour.

Si vous utilisez une machine virtuelle pour l’utilisation de Linux, veillez à ce que la connexion entre votre propre système d’exploitation et votre VM soit établie avec votre carte réseau **filaire** et non avec votre carte WI-FI (par exemple).

Lorsque tout cela est prêt, nous pouvons alors alimenter notre carte et voir l’exécution du système sur Hercules.

**Utile :** Un tutoriel est disponible sur le wiki d’Armadeus. Celui-ci sera très utile pour connaître les commandes importantes et la configuration de votre carte. La carte Armadeus utilisée est une APF27.

# Boa

Objectif : Comprendre et utiliser les scripts CGI afin d’afficher des programmes personnels dans des pages web.

On vérifie que l’on accède toujours à la page index.html grâce au serveur boa déployé sur la carte lors du tp précédent.



On voit qu’en tapant l’adresse de notre carte Armadeus dans la barre d’URL de notre navigateur, on obtient la page HTML précédemment créé et flashée sur notre carte. Le serveur web est donc bien opérationnel.

Le CGI (Common Gateway Interface) est une interface utilisée par les serveurs http. Au lieu de renvoyer le contenu d’un fichier, le serveur http exécute un programme et retourne le contenu généré.

On s’assure que notre serveur boa peut exécuter des scripts CGI en vérifiant que la ligne « ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/ » est bien décommentée dans le fichier de configuration du serveur web : *$ARMADEUS\_ROOTFS\_DIR/etc/boa/boa.conf*

On va maintenant réaliser un script cgi qui permet d’afficher la sortie de la commande ifconfig dans une page web.

Pour cela, on va commencer par créer un script qui lance la commande *ifconfig*. On placera ce dernier dans *$ARMADEUS\_ROOTFS\_DIR/usr/lib/cgi-bin* et on le nommera *get\_ipconfig.sh*.

#!/bin/sh

echo -e "Content-type: text/html\r\n\r\n";

echo -e `/sbin/ifconfig`;

On donne ensuite les droits à notre script grâce à la commande *chmod a+x get\_ipconfig.sh.*

L’étape suivante consiste à modifier la page index.html afin d’ajouter un lien permettant de lancer le script. On ajoute donc la ligne ***<a href=http192.168.1.16/cgi-bin/get\_ipconfig.sh****>* ***Get the board ip config </a>***dans le body de notre page html. Un clic sur ce lien permettra de lancer le script qui affichera le résultat de la commande ifconfig dans le navigateur.

Un fois cette manipulation réalisée, il ne reste qu’à générer le nouveau système de fichiers de la carte et de la flasher. Une fois cette manœuvre effectuée et la carte bootée, on lance le serveur web avec la commande ***boa***.

Sur notre navigateur, on accède à notre serveur web en entrant l’adresse ip de la carte dans la barre d’URL. Un clic sur le premier lien permet d’exécuter et d’afficher le résultat de notre script.

PHOTO DE INDEX.HTML dans un navigateur

Comme nous avons vu plus haut, nous sommes obligé de lancer le serveur web « boa » manuellement avec la commande. Pour éviter cela, nous pouvons créer un script permettant de lancer le serveur web au démarrage de la carte. Pour cela, nous allons créer un fichier nommé **S90boa** dans le dossier */home/student/Documents/armadeus-5.3/buildroot/output/target/etc/init.d*. Ce script contiendra les lignes suivantes.

#!/bin/sh

echo -e `boa`;

Le nom de ce script est important. En effet, c’est dans ce dossier que se trouvent les scripts qui seront lancés par le système au démarrage de la carte. L’ordre de lancement des différents scripts dépend du préfixe de chaque fichier. Tous les fichiers commencent par S##xxx avec ## un chiffre et les x un nombre de lettres indéfini. Plus les deux chiffres sont proches de zéro, plus le script sera prioritaire. Un script commençant par S20 sera donc lancé avant un script commençant par S50. Nous nommons notre script S90boa de manière à ce que ce soit le dernier script lancé.

**Remarque :** Il est préférable de lancer notre script en dernier afin de s’assurer que toutes les ressources matérielles à notre script soient initialisées.

Au boot de la carte, nous n’avons maintenant plus besoin d’effectuer la commande *boa* pour lancer le serveur web. Il se lancera en effet de manière autonome grâce au script. En regardant un peu les log lors du démarrage de la carte, on peut s’apercevoir que notre serveur s’initialise bien.

Nous allons maintenant afficher notre binaire HelloArm (qui n’a pour rôle que d’afficher une simple phrase) dans notre page web grâce à un script cgi. On commence par ajouter notre binaire HelloArm dans le dossier */home/student/Documents/armadeus-5.3/buildroot/output/target/usr/bin* de manière à ce qu’il soit exécutable par le serveur web boa. On ajoute ensuite un script *makehw.sh* dans le dossier */home/student/Documents/armadeus-5.3/buildroot/output/target/usr/lib/cgi-bin.* Il permettra de lancer le fichier exécutable (binaire) compilé. Ce script sera composé des lignes suivantes :

#!/bin/sh

echo -e "Content-type: text/html\r\n\r\n";

echo -e `/usr/bin/HelloArm`;

Nous modifions ensuite notre page index.html pour ajouter un lien vers ce script. En dessous du lien pour afficher le contenu de la commande ifconfig, nous ajoutons la ligne suivante : *<a href=http://192.168.1.16/cgi-bin/makehw.sh> Get the board helloWorld </a>*

A ce stade, nous avons le schéma d’exécution suivant :

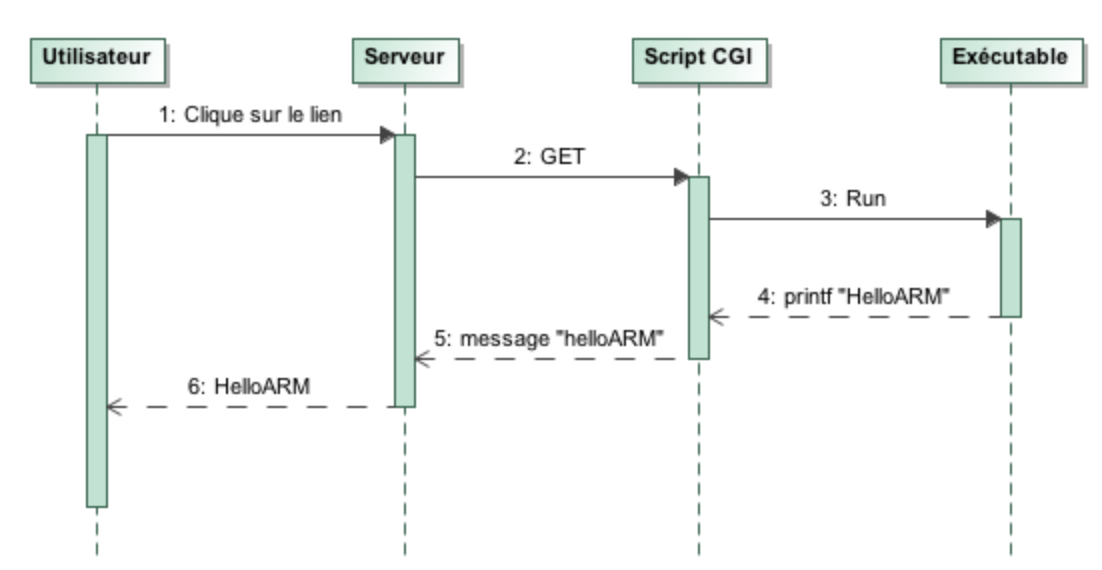


Figure Schéma d'exécution

Il ne nous reste plus qu’à générer le nouveau système de fichiers de la carte et de la flasher. Une fois la carte démarrée, on peut accéder au serveur web depuis notre navigateur.

PHOTO de HELLOARM dans un navigateur

# Capteur – développement

Tout au long de ce chapitre, nous allons utiliser ce que nous avons fait précédemment en rajoutant un capteur de température et d’humidité. Ce capteur (TI HDC1000) communique grâce au protocole I2C.