Système Embarqué de Type Nœud IoT Communicant Sans Fil

V. Frick[ab], F. Imbert[b]

[a] ICube et pôle CNFM Migrest, Université de Strasbourg, Strasbourg, France
[b] IUT de Haguenau, Université de Strasbourg, Haguenau, France
Contact email : vincent.frick@unistra.fr

Cet article présente la réalisation d'un système embarqué remplissant la fonction de nœud IoT (Internet of Things) mené dans le cadre d'une Situation d'Apprentissage et d'Évaluation (SAÉ). Les étudiants conçoivent un système chargé de mettre en œuvre un capteur environnemental (température, humidité, etc.) numérique ayant un protocole non-conventionnel. Les données du capteur sont gérées par un processeur embarqué et transmises au moyen d'un module de communication radiofréquence. En outre, le système exploite le protocole de communication MQTT pour communiquer avec des clients mobiles. Les étudiants montrent un fort intérêt pour ce projet pluridisciplinaire, qui comporte également une partie dédiée au développement d'une application mobile particulièrement ludique. Les résultats obtenus montrent par ailleurs qu'ils atteignent globalement les objectifs et valident les compétences attendues.

I. Introduction et contexte

Dans le cadre de la formation au BUT de Génie Electrique et Informatique Industrielle les étudiants sont placés dans des Situations d'Apprentissage et d'Évaluation (SAÉ). En particulier, pour les étudiants suivant le parcours Électronique et Systèmes Embarqués (ESE), le sujet d'une des SAÉ de deuxième année porte sur la conception et la réalisation d'un système capable de mettre en œuvre des capteurs et d'en exploiter les données, tout en remplissant la fonction de nœud communicant sans fil de type IoT (Internet of Things).

Les ressources associées à cette SAÉ seront principalement les modules d'enseignement ElSpé 3 (Langages de description matériel), ElSpé 4 (Communication par signaux numériques), Info (notamment la programmation en langage C), ainsi que les documentations techniques des matériels mis en œuvre.

A l'issue de la SAÉ, les étudiants, qui réalisent leur travail en autonomie, doivent être en mesure de démontrer qu'ils ont acquis les compétences attendues en conception, vérification et implantation.

La section II de cet article présente le cahier des charges du système « nœud IoT ». La section III est consacrée à la présentation des ressources matérielles et logicielles. La section IV décrit la mise en œuvre et le déroulement des séances de cette SAÉ. Enfin, la section V expose les modalités d'évaluation, le retour d'expérience et conclut cet article.

II. Cahier des charges du système

Fonctions du système embarqué

Le système à concevoir doit, d'une part, servir à collecter des données environnementales (température, humidité, attitude, luminosité, …) issues de divers

Article publié par EDP Sciences et disponible sur le site https://www.j3ea.org ou https://doi.org/10.1051/j3ea/20241007

capteurs et de les exploiter directement sur une carte FPGA (sélection du mode de fonctionnement et d'affichage : bargraphe par leds, 7 segments, …) et à les transmettre sans fil vers un support mobile (smartphone, tablette, PC client, etc.). D'autre part, il doit être capable de recevoir des commandes envoyées depuis le même support mobile pour déclencher des actions (contrôle de leds, affichage 7 segments, acquittement d'alerte de seuil de capteur, déverrouillage, …).

Si la description des fonctionnalités du nœud se situe volontairement à un niveau d'abstraction relativement élevé, c'est pour laisser un degré de liberté suffisant et inciter les étudiants à exprimer leur créativité et être force de proposition. Il demeure néanmoins que certains aspects du cahier des charges, ainsi que le matériel, sont imposés.

Architecture globale du système

La Fig.1 montre l'architecture globale du système de type « nœud IoT ». Le nœud est composé d'un système sur puce. Les étudiants ont un large degré de liberté concernant le partitionnement matériel/logiciel et l'implantation du système. Cependant, il doit obligatoirement comprendre au moins un bloc matériel dédié, écrit en langage de description matériel (VHDL). Ce bloc est imposé par le cahier des charges pour amener les étudiants à démontrer leur compétence en conception de circuits numériques. Il peut par exemple s'agir d'un module chargé de la réception, de la mise en forme et du traitement des données de capteurs.

En outre, le système contient un processeur de type « soft-core » (NIOS II) programmé en langage C, ainsi que divers capteurs (température, humidité, attitude, luminosité, …) et des modules de communication sans fil de type Wi-Fi (ESP32) et Bluetooth (HC 05).

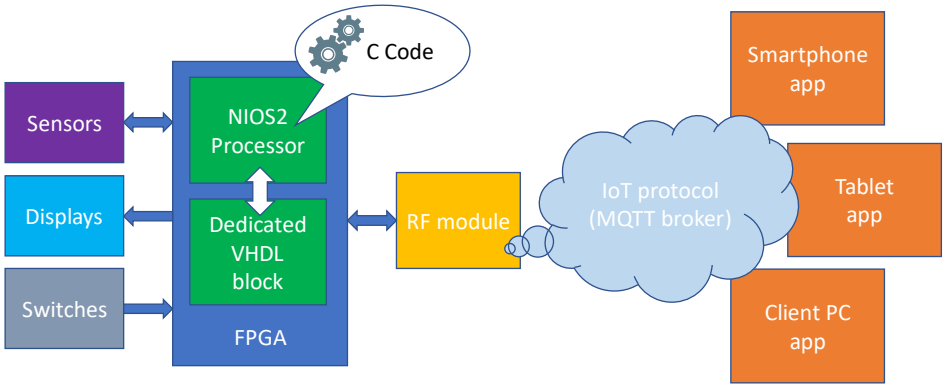

Fig.1. Architecture du Système embarqué de type Nœud IoT communicant sans fil.

III. Ressources matérielles et logicielles

Cartes électroniques

Si le choix de la mise en œuvre, et notamment le partitionnement matériel/logiciel des fonctionnalités du système, est laissé en grande partie à l'appréciation des étudiants, les ressources matérielles mis à disposition pour la réalisation de ce projet sont quant à elles imposées. En l'occurrence, il s'agit :

- d'une carte de développement DE10-Lite terasIC comportant un FPGA de la série MAX10 produit par Intel® FPGA, un port Arduino, un port GPIO, des afficheurs, leds, etc. (1),
- d'une carte Shield comportant un capteur de température et d'humidité (DHT11), des boutons poussoirs, un capteur de luminosité, un buzzer ainsi que des leds. Cette

carte est connectée sur les ports Arduino de la carte DE10-Lite. La particularité decette carte est son capteur de température et d'humidité, qui fonctionne selon unprotocole propriétaire (2). Aucune information *a priori* n'est fournie aux étudiants qui doivent donc préalablement étudier sa documentation et développer le pilotead-hoc afin de le mettre en œuvre,

- d'une carte fille RFS2 comportant divers capteurs ainsi que les modules de communication RF(1) . Cette carte est connectée sur le port GPIO de la carte DE10-Lite.

Le nœud IoT résultant de l'assemblage de ces trois cartes est montré dans Fig.2.

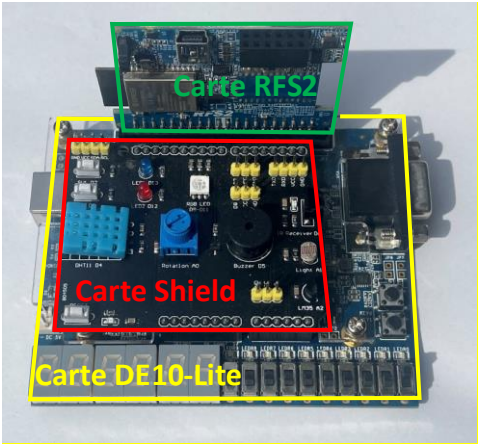

Fig.2. Photo du nœud IoT complet.

Logiciels de développement

Afin de concevoir, simuler, synthétiser et mettre en œuvre le système, les étudiants ont besoin des outils logiciels dédiés, notamment ceux associés aux matériels Intel® FPGA et terasIC, mais également pour le développement d'applications mobiles. Il s'agit en l'occurrence :

- du logiciel de simulation de circuits numériques ModelSim® ;
- de la suite de développement Quartus® pour la synthèse et l'implantation de systèmes numériques sur cible Intel® FPGA ;
- du logiciel Eclipse, couplé à l'environnement Quartus®, pour le développement du programme en langage C qui permet de mettre en œuvre le processeur embarqué NIOS II.
- du logiciel WinDev® qui permet de développer des applications pour appareils mobiles de type smartphone ou tablette (5).

Base de projet

Outre les ressources matérielles et logicielles, une base de projet Quartus dédiée à cette SAÉ est également mise à disposition des étudiants. Cette base de projet comporte une cellule décrite en langage VHDL que l'on nommera SAE4_TOP (SAE4_TOP.vhd). Elle constitue le niveau hiérarchique principal du système. L'architecture de cette cellule est amenée à évoluer au cours du projet par l'ajout progressif de fonctionnalités (blocs numériques décrits en VHDL, ports microcontrôleurs sur le processeur, etc.). Les ports de SAE4_TOP sont quant à eux déjà assignés à toutes les différentes entrées/sorties

disponibles sur la carte DE10-Lite (concept de « Golden Top »). Cette approche permet d'éviter les écueils pour ce projet qui est par ailleurs d'un niveau de complexité relativement avancé pour le niveau attendu en deuxième année du BUT GEII (BUT2), en raison de la variété des notions qu'il convoque (co-design matériel/logiciel, développement de driver matériel, protocoles de communication sans fil, développement d'application mobile).

Dans le même esprit, la base du projet comporte également une base de cœur de processeur NIOS II, un contrôleur de SDRAM et un bloc PLL chargé de générer les horloges adaptées pour l'utilisation de la SDRAM par le processeur. Les parties les plus complexes liées aux horloges, au processeur NIOS et au contrôleur de mémoire SDRAM sont préconfigurées (Fig.3).

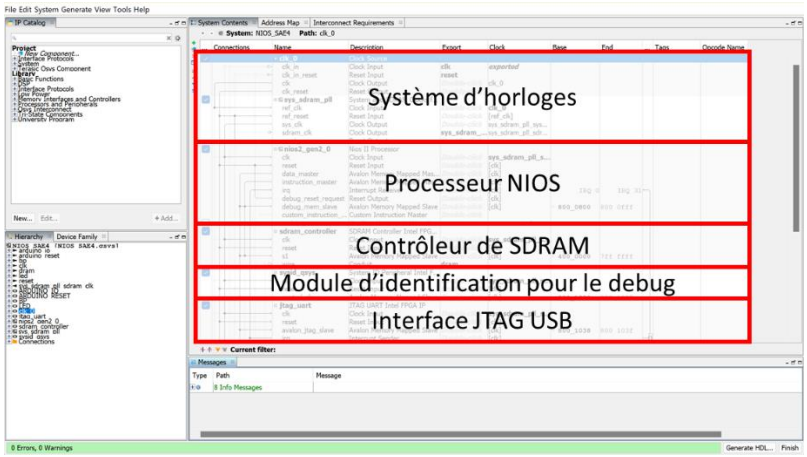

Fig.3. Vue de l'interface Platform Designer permettant de créer des systèmes embarqués avec cœur de processeur NIOS II sous Quartus.

Les notions d'architectures des processeurs n'étant pas au programme de BUT2, l'idée est de garantir que chaque étudiant construit son système sur une base saine. Aussi, une partie de la première séance encadrée, dédiée au démarrage de la SAÉ, est consacrée à la présentation de cette base et à l'initiation à l'outil Platform Designer, intégré à Quartus. Cet outil, qui sert à construire des systèmes embarqués complexes, est ici uniquement utilisé pour rajouter des fonctions et ports simples à configurer (contrôleur I2C, ports d'entrées/sorties microcontrôleur, etc.). Les étudiants sont alors affranchis de la configuration des blocs complexes dépassant leur niveau de compétences (paramétrage du cœur de processeur, dimensionnement de la mémoire cache, paramétrage des vecteurs de reset et d'interruptions, timings de la mémoire SRAM associée, etc.).

IV. Déroulement des séances

La durée totale du projet est de 96 heures qui se répartissent en 14 séances encadrées et 10 séances non-encadrées. Le projet se décompose en trois grandes phases. Chaque étudiant travaille individuellement et dispose de l'équipement nécessaire à la réalisation du projet.

Mise en œuvre du capteur de la carte Shield

Cette première phase porte sur le développement de la couche matérielle destinée à mettre en œuvre le capteur de la carte Shield.

Tout d'abord, les étudiants apportent les modifications nécessaires au code VHDL de la cellule de niveau hiérarchique principal SAE4_TOP de manière à pouvoir interfacer la carte Shield sur la carte mère DE10-Lite. Cette manipulation est essentielle pour permettre la communication avec les organes de la carte Shield et notamment avec le DHT11. Par défaut tous les ports d'entrées/sorties ARDUNIO (et GPIO) de la cellule SAE4_TOP sont pendants et forcés en haute impédance. Selon le choix technique de développement du driver du DHT11, les étudiants doivent donc relier ces ports soit vers les ports d'un bloc décrit en langage VHD s'ils optent pour une version matérielle, soit à des ports du système NIOS s'ils optent pour une version logicielle. Dans le deuxième cas, cela s'accompagne également par une modification du système NIOS à l'aide de l'outil Platform Designer. En effet, ils doivent rajouter les ports microcontrôleurs ad-hoc pour établir le lien entre le NIOS et la carte Shield.

Par la suite, avant de développer le driver à proprement parler, les étudiants doivent préalablement analyser le protocole de communication ainsi que les timings du signal du capteur DHT11. La communication de ce dernier avec l'hôte s'effectue via un seul bit, sur un port bidirectionnel (Fig.4). Le protocole implique l'envoi d'un '0' par l'hôte pendant au minimum 18ms (2). Puis l'hôte se met en mode écoute et le DHT11 envoie sa trame de données en réponse.

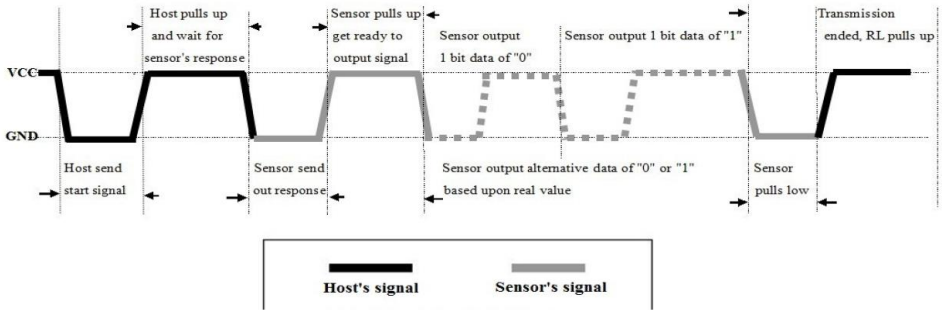

Fig.4. Protocole de communication unifilaire du capteur DHT11 (2).

La prochaine phase de conception est alors le développement du pilote, en VHDL ou en langage C, permettant de communiquer et de collecter les données du capteur DHT11.

Enfin, les étudiants développent un bloc matériel en VHDL chargé, d'une part de manipuler les données du DHT11, et d'autre part de permettre la sélection du mode d'affichage directement sur les interfaces lumineuses de la carte DE10-Lite (afficheurs 7 segments, bargraphe par leds). Notons ici que ce bloc est indépendant du driver et que son développement en langage VHDL est imposé pour garantir que tous les étudiants ont conçu au moins une entité avec une complexité suffisante pour évaluer leur niveau de compétence dans ce langage.

Remarquons également que tous les développements en VHDL sont menés à l'aide de l'outil ModelSim® et doivent obligatoirement faire l'objet de simulations préalables, ceci afin de forcer au développement de méthodologies de conception rigoureuses.

Mise en œuvre de la carte RFS2

La deuxième phase porte sur la mise en œuvre de la carte RFS2. D'un point de vue matériel, le système est modifié de manière à pouvoir exploiter les modules radio (Wi-Fi

ou Bluetooth) à partir de la carte DE10-Lite. Il s'agit essentiellement de rajouter une interface UART sur le système NIOS et de configurer les ports GPIO.

D'un point de vue logiciel, les étudiants écrivent un programme permettant de relier le système à un point d'accès sans fil. Dans le cas du Wi-Fi, option majoritairement choisie par les étudiants car plus simple à mettre en œuvre, un boîtier Raspberry Pi® est utilisé comme serveur d'un réseau local. Ceci permet de simplifier la procédure de connexion, plus complexe dans le cas d'un réseau sécurisé comme Eduroam par exemple. Notons que le boîtier est également utilisé comme broker pour le protocole MQTT décrit ci-dessous.

Protocole MQTT et développement de l'application mobile

La dernière phase est exclusivement logicielle. Elle porte sur le développement du programme permettant l'émission et la réception de données sur le système afin de créer un nœud IoT. Le principe de messagerie employé est basé sur le protocole MQTT (3). Dans le cas d'une configuration en Bluetooth, cela impose la création d'une passerelle vers le protocole MQTT. Dans le cas d'une configuration Wi-Fi, la procédure est simplifiée notamment en raison de la prise en charge native du protocole MQTT par le processeur ESP32 présent sur la carte RFS2.

Le principe MQTT est basé sur l'utilisation de thèmes, ou « topics », créés au niveau du broker, auxquels les clients (le nœud IoT et les terminaux, comme un PC ou un smartphone par exemple) doivent s'abonner pour recevoir et envoyer des messages (3). Les messages publiés peuvent contenir soit des données du capteur, soit des commandes. Dans le cadre de ce projet, deux topics ont été utilisés : l'un pour diffuser les données du capteur vers les terminaux abonnés, l'autre pour envoyer des commandes depuis les terminaux vers le nœud IoT. Les commandes permettent en l'occurrence principalement de changer le mode d'affichage sur la carte DE10-Lite et de choisir le type de grandeur mesurée (température, humidité, etc.). Afin de tester la communication, les étudiants utilisent une application dédiée au protocole MQTT permettant d'envoyer et de recevoir les paquets en ligne de commande, telle que MQTT.fx par exemple (4).

La dernière étape consiste à créer une application mobile associée au nœud IoT. Pour ce faire, les étudiants utilisent le logiciel WinDev® Mobile, dont l'interface et le mode de programmation graphique très intuitifs permettent de développer facilement des applications pour équipements fonctionnant sous environnement Android ou IOS (Fig.5).

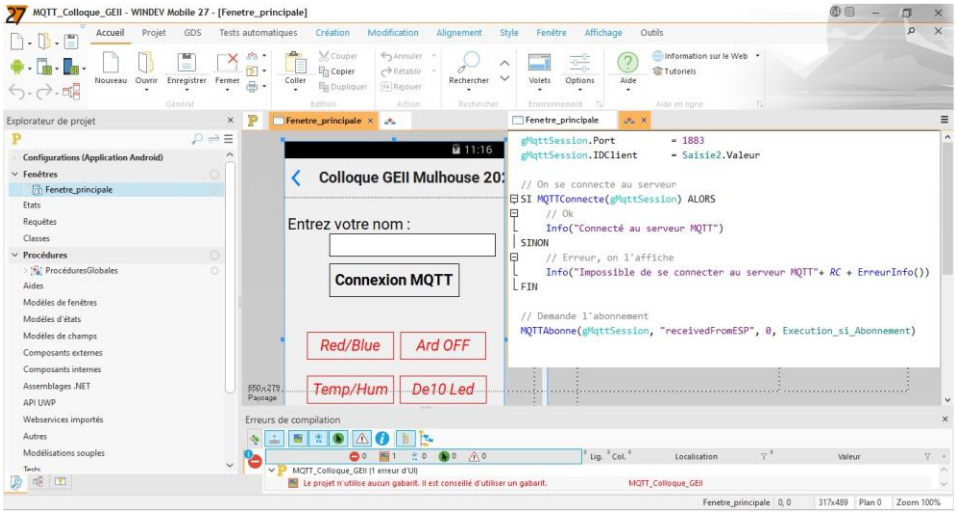

Fig.5. Développement de l'application mobile du projet à l'aide de WinDev® Mobile.

V. Evaluation, retour d'expérience et conclusion

Le projet est évalué sur la base de trois items : 1) la fonctionnalité et la qualité des livrables (matériel et logiciel), 2) une démonstration du fonctionnement du système, au cours de laquelle chaque étudiant doit être en mesure d'expliquer en détail ses choix techniques, 3) un rapport écrit. La durée de la démonstration est d'environ un quart d'heure par étudiant. A travers les thèmes abordés : langage de description matériel, mise en œuvre de FPGA et objets connectés, le projet sert à évaluer trois compétentes principales du parcours ESE du BUT GEII : la conception, la vérification et l'implantation.

Dans l'ensemble, le groupe d'étudiants (une quinzaine en parcours ESE) est très enthousiasmé par le projet, notamment pour son côté pluridisciplinaire. Il apparaît clairement que l'aboutissement à une application mobile donne un aspect ludique qui rajoute à la l'intérêt porté pour cette SAÉ. Les premières expériences montrent néanmoins que le volume horaire est tout juste suffisant pour mener le projet à bien dans sa globalité. Ceci provient essentiellement de deux raisons : d'une part, les étudiants passent du temps à comprendre le protocole du DHT11 et à en développer le pilote. En particulier, les étudiants qui font le choix de développer un pilote logiciel sont souvent confrontés à des problèmes de timing des signaux. En cela, le projet atteint ses objectifs pédagogiques puisqu'ils prennent alors conscience de la notion de temps réel et de temps de cycle machine qui n'est pas évident à appréhender en programmation de microcontrôleurs. Par ailleurs ils gagnent en méthodologie de travail et prennent notamment conscience du gain significatif à simuler le fonctionnement d'un circuit numérique afin de le mettre au point, préalablement à sa synthèse et à son implantation sur la cible matérielle.

Au regard des résultats présentés par les étudiants, l'expérience semble très concluante. Elle a d'ailleurs fait l'objet d'une démonstration d'un nœud IoT réalisé par un étudiant à l'occasion du 49ème Colloque GEII, à Mulhouse, en juin 2023.

Remerciements

Les auteurs veulent remercier le GIP-CNFM, le laboratoire ICube et le laboratoire IPHC pour leur soutienfinancier à la création de cette plateforme. Remerciement également à France 2030" et le projet "ANR-23-CMAS-0024 INFORISM.

Références

1. https://www.terasic.com.tw/
2. https://www.electronicoscaldas.com/datasheet/DHT11_Aosong.pdf
3. https://mqtt.org/
4. https://softblade.de/en/welcome/
5. https://pcsoft.fr/