Ce qu'il faut savoir...

Depuis quelques années, les véhicules routiers et agricoles sont équipés de série d'un bus numérique qui relie tous les organes électroniques : l'injection électronique, le tableau de bord, l'ABS, le radar de recul, ... Pouvoir s'interfacer avec un bus CAN automobile, c'est pouvoir récupérer toutes les informations utiles de votre voiture comme le niveau de carburant, le nombre de tour/minute, la vitesse du véhicule, ...

- CPU ARM7 S3C44B0X 60MHz,
- 16 Mo de SDRAM,
- 2 Mo de NOR flash,
- 16 Mo de NAND flash,
- Ethernet 10 Mbits/sec,
- 2 ports COM série,
- 1 port USB host v1.1,
- Bus I2C 400KHz,
- Interface LCD CSTN 320x240,
- uClinux 2.4.24.

Notre carte électronique dispose d'un processeur performant mais fonctionnant à une fréquence raisonnable : 60MHz. Aucun besoin d'utiliser une carte à 200 ou 400 MHz pour notre systèmes même si celui-ci doit communiquer avec plusieurs périphériques et transmettre des données au travers d'un modem GPRS.

En fait, si l'on examine bien les éléments qui composent notre système (GPS, modem GPRS, interface CAN, contrôleur vidéo LCD, ...) on remarque que presque toutes les fonctionnalités de notre application sont réalisées automatiquement par un organe de contrôle spécifique :

- *GPS* : un module spécifique se connectera à notre contrôleur USB host afin de fournir les coordonnées géographiques et l'heure absolue. Le module GPS choisi comprends en réalité un convertisseur RS232 / USB, tel que les composants FTDI. Ceci nous permettra donc de récupérer toutes les données comme si nous nous connections au travers d'un port série (*/dev/ttyUSB0* par exemple),
- *Modem GPRS* : nous utiliserons un modem piloté par des commandes AT envoyées au travers d'un lien série asynchrone. Ce modem ne sera pas connecté au réseau téléphonique commuté (RTC) mais au réseau GPRS, qui nous offrira une passerelle vers le monde Internet. Nous obtiendrons alors une adresse IP et une connexion IP sortante (pour des raisons de sécurité). À l'aide de quelques commandes AT bien choisies, nous serons en mesure de transmettre des données par lien série, qui seront alors transmises à un serveur distant,
- *Contrôleur de bus CAN* : le bus CAN est un bus bien particulier auquel il faut s'interfacer au travers d'un contrôleur spécifique. Ce contrôleur nécessite un driver adapté bien entendu, qui sera en charge de récupérer des octets issus du bus CAN en fonction de l'interface du CPU choisi (bus série, parallèle, DMA, ...),
- *Contrôleur vidéo LCD* : certains processeurs tels que celui que nous avons choisi (lS3C44B0X) disposent d'un contrôleur LCD intégré qui permet de créer un lien direct entre une partie de la mémoire SDRAM (mémoire vidéo) et un LCD externe. Un transfert DMA permanent (donc sans sollicitation du CPU) permettra de transférer tout le plan de la mémoire vidéo vers l'écran du LCD.

Comme vous avez pu le constater, la plupart des fonctionnalités de notre système sera assurée par un périphérique hardware. Le CPU et l'application développée pour le noyau uClinux 2.4 tiendra alors le rôle de chef d'orchestre, en charge de piloter les différents périphériques et de prendre les décisions qui s'imposent.

Un CPU sans MMU (gestion de la mémoire virtuelle) à 60 MHz et un noyau uClinux adapté suffisent donc amplement. Nous aurions pu prendre une plate-forme à base de CPU avec MMU à 100MHz et le noyau Linux 2.6. Ces 2 configurations correspondent très certainement aux 2 configurations minimales. Une plate-forme de puissance inférieure serait probablement mise en difficulté pour assurer toutes les fonctions requises.

Bus CAN v2.0B

Le bus CAN a été conçu par la société BOSCH dans les années 80 pour les véhicules automobiles qui comportait de plus en plus d'équipements électroniques. Le but était de pouvoir connecter entre eux les différents calculateurs du véhicule : injection électronique, airbag, ABS, radar de recul, tableau de bord, ... C'est finalement la norme ISO11898 qui définit précisément le cadre de fonctionnement du bus CAN.

Le bus CAN est composé d'une paire torsadées dans la plupart des cas : CANL et CANH. Cette paire de conducteur et la couche réseau CAN permettent de s'affranchir des parasites électromagnétiques. La convention est d'offrir une interface CAN composée de 4 fils :

- GND,
- VCC,
- CANL,
- CANH.

... afin de pouvoir mettre les différentes cartes électroniques connectées au même potentiel (GND) et d'alimenter certaines par la même occasion (VCC).

Le bus CAN pourrait être comparé au bus RS422 qui, lui aussi, utilise une paire différentielle pour transmettre les données, mais il n'en est rien. Le bus RS422 permet une protection relativement faible mais efficace contre les perturbations électromagnétiques (de part la paire différentielle) mais qui n'est pas suffisamment performante pour être intégrée dans un milieu extrêmement perturbé tel que celui d'un véhicule.

La caractéristique essentielle du bus CAN est donc sa grande immunité au bruit, immunité qui est assurée par l'emploi d'une paire différentielle et par l'élaboration d'une électronique capable de corriger automatiquement les erreurs de transmission, ou bien, si la correction s'avère inefficace, de redemander l'émission de la trame en erreur.

Ceci lui permet d'atteindre des débits élevés sur des distances relativement importantes :

- 1Mbit/s sur 40m,
- 500 kbit/s sur 100m,
- 100 kbit/s sur 500 m,
- 20 kbit/s sur 1000m.

La trame CAN n'est pas un protocole point-à-point, bien au contraire : un message est diffusé avec un identifiant particulier à tous les équipements présents sur le bus CAN. Ainsi, il convient à chacun de savoir si la trame détectée lui est destiné en examinant l'identifiant. L'identifiant n'est donc pas un numéro de destinataire mais plutôt un thème de discussion, comme pour dire *voici les données correspondant aux éléments de sécurités de l'habitacle*. Ce thème est donc accessible par tous, et seuls

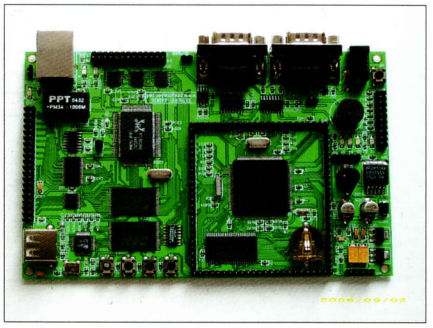

Figure 1. Carte de développement EUROP-44BOX