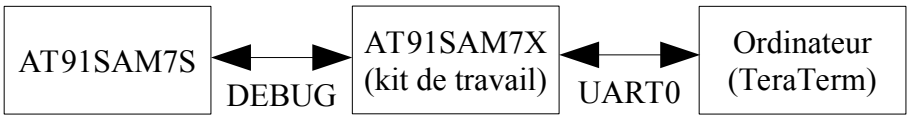
Microprocesseur

- Projet -
Gestion de flux de données

Le principe du projet de cette année est de programmer un système capable de gérer un flux entrant de données. L'affichage se fera par l'intermédiaire de la liaison UART et du terminal virtuel TeraTerm. Le flux de données entrant arrivera par la liaison DEBUG. Cette liaison utilise le même principe que la liaison UART série, mais en utilisant une architecture simplifiée. A notre niveau, elles sont équivalentes (mêmes noms de registres, mêmes masques...). Seuls quelques champs de registres ne sont pas implémentés, vérifiez que vous n'en avez pas besoin. Notamment le module DEBUG n'offre pas la possibilité d'*oversampling* (bit over=1 dans le registre US_MR).

1) Communications

La carte est à connecter selon le schéma suivant :



Les paramètres de communication entre le kit AT91SAM7X et l'ordinateur sont libres (mais devront être justifiés). Pour la communication entre les deux kits, ils sont les suivants:

- 200 kbps
- 8 bits par mot
- 1 bit de stop
- pas de parité
- pas de contrôle de flux

Le kit AT91SAM7S répond aux commandes suivantes :

- Toutes les secondes, il envoie le texte "allo?↵"
- Lorsqu'on lui envoie le caractère '**+**', le texte envoyé périodiquement est modifié.
- Lorsqu'on lui envoie le caractère '**?**', il répond un texte plus conséquent, puis arrête de communiquer pendant 5 secondes
- Lorsqu'on lui envoie le caractère '**s**', il envoie une image en ASCii Art (caractères qui représentent une image). Ce transfert se fait en une seule fois. Lorsque l'envoi est effectué, le système arrête de communiquer pendant 20 secondes. La taille dede cette première image est de 3167 caractères.
- Lorsqu'on lui envoie le caractère '**L**', il répond une autre image en ASCii Art, mais cette fois, les caractères sont envoyés dans le désordre. Pour chaque caractère de l'image, le système envoie d'abord le caractère *255*. Ensuite, il envoie un numéro de colonne puis un numéro de ligne (codés sur un octet chacun). Finalement, le kit envoie le caractère à afficher.
  Dans ce mode, l'envoi de chaque caractère se fera donc sous la forme :
  <255><colonne><ligne><caractère>

Chaque caractère est à afficher au fur et à mesure de la réception. Il y a au total 7200 caractères répartis sur une matrice de 150 colonnes et 48 lignes. Une fois l'envoi terminé, le kit arrêtera d'envoyer "allo?↵" pendant 10mn

- la commande ' ' (espace), force le kit à recommencer les appels "allo?↵"

2) Étapes de construction du projet

a) Réception de données / affichage

Dans un premier temps, vous devez afficher les données qui sont envoyées par le kit AT91SAMS sur le terminal de l'ordinateur. Pour cela, il "suffit" de renvoyer vers le terminal tous les caractères qui arrivent depuis le kit AT91SAM7S. Il sera nécessaire de gérer un débit plus élevé en réception qu'en émission puisque l'ordinateur ne peut pas dépasser 115 200bps sur la liaison UART.

b) Communication bidirectionnelle

Dans un second temps, lisez les caractères frappés dans le terminal. Si un caractère correspond à une commande, renvoyez le au kit AT91SAM7S, sinon, affichez 'erreur' sur le terminal. Cette partie ne doit pas impacter le renvoi des messages effectué pour la partie a).

Si cette partie fonctionne, vous devriez être en mesure d'obtenir une réponse cohérente aux instructions '**?**', '**+**' et '**s**'.

c) Affichage progressif

Cette partie consiste à gérer l'affichage de la commande '**L**'. Une fois que vous avez transmis le caractère 'L', vous êtes désormais susceptibles de recevoir des caractères non affichables (code ASCii inférieur à 32 ou 0x20). Le caractère *255* est toujours suivi de deux caractères pour indiquer les nouvelles coordonnées. Il faudra décoder le contenu de ces trames pour positionner correctement le curseur. Tous les autres caractères sont affichables. Si rien n'est reçu pendant plus de deux secondes, c'est que l'animation est terminée, il faut repositionner le curseur en haut à gauche.

3) Précisions techniques

a) Conception d'une FIFO (au cas où...)

Pour réaliser une FIFO, il faut au moins un tableau qui contienne les caractères, un indice pour repérer la case d'écriture et un autre pour la case de lecture. Une écriture se fait en affectant la case d'écriture, puis en incrémentant son indice modulo le nombre d'éléments dans le tableau. Une lecture s'effectue avec les mêmes étapes : lecture de l'élément dans le tableau puis incrément de l'indice. Si les indices d'écriture et de lecture sont identiques, la FIFO est soit pleine, soit vide. À vous de choisir comment vous préférez gérer ces cas de figure (on peut par exemple interdire de remplir complètement la file pour clarifier la situation). Le plus pratique est de réaliser des fonctions de lecture et d'écriture qui allègent le code C et la compréhension du programme.

b)gestion du curseur

Pour positionner le curseur à un endroit déterminé, la séquence suivante doit être envoyée au terminal :

- un caractère de valeur 0x1B (échappement)
- le caractère '['
- le numéro de ligne au format texte : un ou plusieurs caractères entre 0x30 ('0') et 0x39 ('9'), par exemple, la ligne 14 est référencée par le caractère '1' puis le caractère '4'.
- le caractère ';'
- le numéro de colonne au format texte : un ou plusieurs caractères entre 0x30 ('0') et 0x39 ('9')
- le caractère 'H'

ATTENTION, la ligne (position y) est envoyée avant la colonne (position x) pour des raisons techniques historiques. La numérotation se fait à partir de (1,1)

La séquence abrégée 0x1B puis '[' et 'H', permet de revenir en haut à gauche du terminal.

4) Évaluation

L'évaluation dépend (par ordre d'importance) : du rapport (qualité et contenu), du respect des consignes, de l'avancement du projet ainsi que du niveau technique de sa réalisation, éventuellement les développements effectués en TPs.

a) format technique :

Le rapport est à rendre pour le Lundi 27 Mai à 9h (heure du serveur d'envoi) par e-mail.

- Le sujet de l'e-mail commencera par [MI202]
- Destinataires
  - Le destinataire principal sera votre encadrant :
    - valery.lebret@enseirb-matmeca.fr
    - mathieu.leonardon@u-bordeaux.fr
    - yannick.bornat@enseirb-matmeca.fr
  - l'adresse bornat@ipb.fr en CC pour archive.
  - Les autres membres du binôme/trinôme en CC (**tous les membres reçoivent donc l'e-mail de soumission du rapport et seront de ce fait considérés solidaires en cas de non respect des consignes**)
- Le document sera au format PDF, 1Mio maximum (1048576 octets).
- Le fichier du rapport sera nommé MI202_XXX_YYY.pdf (où XXX et YYY représentent le nom des membres du binôme).
- Le code sera joint sous forme d'un seul fichier .c nommé MI202_XXX_YYY.c (où XXX et YYY représentent le nom des membres du binôme)

b) contenu attendu

Le rapport n'excédera pas 15 pages. Il contiendra notamment les structures choisies et le rôle qui leur est attribué.

- Le rappel du sujet n'est pas nécessaire.
- Le rapport ne doit pas contenir de code C puisque le code est fourni séparément. Quelques lignes sont tolérées pour expliciter une syntaxe de configuration.
- Le rapport présentera dans un premier temps les choix techniques qui ont été faits, ainsi que les algorithmes utilisés, et quelles fonctionnalités ont été choisies pour être implémentées en

interruption.

- Pour chaque fonctionnalité, vous devez justifier pourquoi elle a été implantée en interruption ou en scrutation.
- Si plusieurs sources d'interruptions sont actives en même temps, il faudra argumenter l'attribution de leur priorité.
- Si vous utilisez des machines d'état, il sera intéressant de les représenter.
- Vous aurez besoin d'utiliser au moins un tableau pour du stockage temporaire, sa taille (leur taille s'il y en a plusieurs) doit être argumentée.
- Un changement de codage sera nécessaire pour la partie 2c), détaillez bien l'algorithme.
- La configuration des modules matériels sera expliquée en détail (pourquoi telle valeur dans tel registre, et ce que ça implique au niveau du système). Si des valeurs utilisées sont le résultat de formules, elles sont à expliciter.

Toute remarque effectuée par votre encadrant est prioritaire sur ce sujet.