Protocole de communication Microcontrôleur-RaspberryPi

Introduction

Le protocole de communication défini dans ce document permet à deux programmes de communiquer via une liaison série. L'un des deux appareils communiquant est considéré comme le maitre, et l'autre est l'esclave. Les trames envoyées par l'esclave sont toujours des réponses à une précédente trame reçue de la part du maitre, toutefois le moment de la réponse de l'esclave n'est pas toujours connu à l'avance par le maitre.

Format des trames binaires

Type de trame	Longueur	ID conversation	ID ordre	Données	Checksum
1 octet	1 octet	1 octet	0 ou 1 octet	0 à 251 octets	1 octet

Type de trame

Valeur	Nom	Description	Expéditeur
0xFF	NEW_ORDER	Nouvel ordre dont l'exécution nécessite du temps	Maitre
0xFE	END_ORDER	Fin d'un ordre dont l'exécution nécessite du temps	
0xFD	VALUE_REQUEST	Nouvelle requête à réponse immédiate	
0xFC	EXECUTION_BEGIN	Début de l'exécution d'un ordre long	Esclave
0xFB	EXECUTION_END	Fin de l'exécution d'un ordre long&	
0xFA	STATUS_UPDATE	Information concernant l'ordre en cours d'exécution	
0xF9	VALUE_ANSWER	Réponse immédiate	
Autre	-	Il ne s'agit pas d'un début de trame valide	-

ID ordre

Valeur entre 0x00 et 0xFF, représentant un ordre que le maitre est susceptible de donner à l'esclave. La correspondance « ID – Action » doit être définie au préalable et connue des deux interlocuteurs. Ce champ n'est présent que dans les trames de type NEW_ORDER et VALUE_REQUEST.

ID conversation

Identifiant de la conversation. Compris entre 0x00 et 0xFF. Le maître choisi un identifiant de conversation libre au moment d'envoyer une trame NEW_ORDER ou VALUE_REQUEST, ceci marque le début d'une nouvelle conversation, l'identifiant choisi est alors occupé et ne peut plus être utilisé pour une autre conversation jusqu'à sa libération. Cet identifiant sera utilisé dans chaque trame de la conversation.

Checksum

Somme des octets des champs 'Type de trame', 'Longueur', 'ID conversation, 'ID ordre' et 'Données'.

Données

La taille de ce champ peut aller de 0 à 250 octets. Le format des données contenues doit être spécifié au préalable pour chaque 'ID ordre' et connu des deux interlocuteurs.

Longueur

Longueur totale de la trame, pouvant aller de 4 à 255. La longueur inclut l'intégralité des champs de la trame.

Déroulement d'une communication

La communication est toujours initiée par le maitre, l'esclave ne peut que lui répondre et ne peut donc pas émettre spontanément. 'ID' désigne ici l'identifiant de conversation.

Légende :

	Message attendant un acquittement. Si aucune trame d'acquittement (ayant le même ID que ce message) n'est reçue après un temps TIMEOUT défini en fonction de la vitesse de transmission, alors ce message sera renvoyé à l'identique.
	Message d'acquittement. Pour chaque trame reçue attendant un acquittement, un message de ce type sera envoyé (et ce même en cas de doublons, mais dans ce cas l'ordre reçu en double ne doit être exécuté qu'une seule et unique fois)
	Message optionnel qui ne fera l'objet d'aucun acquittement

Ordre long

Maitre	→ [NEW_ORDER] [ID] [orderID] [data] →	Esclave
	← [EXECUTION_BEGIN] [ID] [data] ←	
	← [STATUS_UPDATE] [ID] [data] ←	
	← [EXECUTION_END] [ID] [data] ←	
	→ [END_ORDER] [ID] [data] →	

Requête à réponse immédiate

Maitre	→ [VALUE_REQUEST] [ID] [orderID] [data] →	Esclave
	← [VALUE_ANSWER] [ID] [data] ←	

Comportement en cas d'erreur de transmission

Chaque trame est associée à une conversation via son identifiant de conversation, noté ID.

On définit également une gradeur TIMEOUT. Il faut que la durée TIMEOUT soit suffisante pour envoyer une trame et recevoir l'acquittement lui correspondant. Ainsi, si au bout de cette durée l'acquittement n'a pas été reçu, on peut supposer que la trame a été perdue.

Ordre long

Voici le détail des communications effectuées lors de la transmission d'un ordre « long ».

Maitre	Esclave
Choix d'un ID n'étant pas déjà utilisé.	
Envoi d'une requête NEW_ORDER.	
	Réception d'une requête NEW_ORDER. Si elle n'est pas valide *(checksum faux, ordre inconnu, …)* la trame est supprimée.
Si au bout de TIMEOUT aucun message de type EXECUTION_BEGIN portant le bon ID n'est reçu, la requête NEW_ORDER est renvoyée à l'identique.	
	Réception d'une ou plusieurs trame(s) NEW_ORDER valide(s) : envoi d'exactement autant de trame(s) EXECUTION_BEGIN. *(Si l'ID est déjà utilisé, l'ordre n'est pas effectué mais acquitté. Si l'ID est nouveau, il est enregistré et l'ordre est effectué.)*
Réception d'une ou plusieurs trames EXECUTION_BEGIN ayant le bon ID : acquittement réussi.	
Réception d'une, aucune ou plusieurs trames STATUS_UPDATE.	Envoi d'une, aucune ou plusieurs trames STATUS_UPDATE
	Fin de l'exécution : envoi d'une trame EXECUTION_END.
Réception d'une ou plusieurs trame(s) EXECUTION_END ayant le bon ID : réponse d'exactement autant de trame(s) END_ORDER, et libération de l'ID de conversation après 2*TIMEOUT (à compter du dernier envoi de trame END_ORDER)	
	Si au bout de TIMEOUT aucun message de type END_ORDER portant le bon ID n'est reçu, la trame EXECUTION_END est renvoyée à l'identique.
	Réception d'une trame END_ORDER correspondant à une commande dont l'exécution est effectivement terminée : libération de l'ID de la conversation.

Requête à réponse immédiate

Voici le détail des communications effectuées lors de la transmission d'un ordre « à réponse immédiate ».

Maitre	Esclave
Choix d'un ID n'étant pas déjà utilisé.	
Envoi d'une requête VALUE_REQUEST.	Réception d'une requête VALUE_REQUEST. Si elle n'est pas valide *(checksum faux, ordre inconnu, …)* la trame est supprimée.
Si au bout de TIMEOUT aucun message de type VALUE_ANSWER portant le bon ID n'est reçu, la requête VALUE_REQUEST est renvoyée à l'identique.	Réception d'une ou plusieurs trame(s) VALUE_REQUEST valide(s) : envoi d'exactement autant de trame(s) VALUE_ANSWER. *(Si l'ID est déjà utilisé, l'ordre n'est pas effectué mais acquitté. Si l'ID est nouveau, il est enregistré et l'ordre est effectué.)*
Réception d'une ou plusieurs trames VALUE_ANSWER ayant le bon ID : acquittement réussi. Libération de l'ID, 2*TIMEOUT après la réception de la dernière trame VALUE_ANSWER.	2*TIMEOUT après l'envoi de la dernière trame VALUE_ANSWER, l'ID est libéré.

Les différents types d'ordres

Chaque ordre est identifié par un « identifiant d'ordre », celui-ci est toujours présent dans l'entête d'une trame et permettra d'interpréter correctement son contenu.

Les ordres longs et les ordres à réponse immédiate

On distingue deux grandes catégories d'ordres :

Les ordres longs

Il s'agit des ordres dont l'exécution implique des composants mécaniques ou électroniques ayant un temps de réponse plus grand ou du même ordre que la durée TIMEOUT. Ils sont donc acquittés au début de leur exécution, puis une seconde fois lorsque l'exécution est terminée. Leur exécution peut nécessiter un temps arbitrairement long, potentiellement infini. Si une forme de « timeout » doit être implémentée, elle doit l'être au niveau de l'esclave qui garantit alors une durée maximale d'exécution.

Les ordres longs utilisent les types de trames suivantes, et sont identifiés grâce à ça :
NEW_ORDER ; EXECUTION_BEGIN ; STATUS_UPDATE ; EXECUTION_END ; END_ORDER

Les ordres à réponse immédiate

Il s'agit des ordres dont l'exécution se déroule en un temps négligeable devant TIMEOUT. Typiquement, les lectures/écritures de données sont de bons candidats.

Les ordres à réponse immédiate utilisent les types de trames suivantes, et sont identifiés grâce à ça :
VALUE_REQUEST ; VALUE_ANSWER

Les identifiants d'ordres

Chaque ordre est identifié par son « identifiant d'ordre », qui est un entier non signé codé sur 8 bits. Il est donc possible de définir 256 ordres différents. Etant donné que les ordres longs et les ordres à réponse immédiate sont distinguables grâce au « type de trame », il est possible de définir 256 ordres longs et 256 ordres à réponse immédiate, faisant passer le nombre total d'ordres disponibles à 512. Si jamais, pour une raison quelconque, plus de 512 ordres distincts sont nécessaires, il est recommandé de regrouper les ordres par thème, et de placer l'information manquante dans le champ de données, dont la taille est moins limitée.

Vitesse maximale de communication

La vitesse de communication est limitée par les grandeurs suivantes :
BAUDRATE ; TIMEOUT ; STACK_SIZE

BAUDRATE : débit binaire sur le canal de transmission

TIMEOUT : durée supérieure au temps nécessaire pour recevoir un acquittement

STACK_SIZE : nombre d'ordres pouvant être effectués « simultanément ». Sachant qu'un ordre n'est oublié que 2*TIMEOUT après la fin de son exécution, les ordres en question ne sont pas forcément effectués simultanément mais sont présents simultanément dans la mémoire.

Toutes les communications sont limitées par le BAUDRATE, la limite en termes de nombre d'ordres par seconde dépend bien évidement de la taille – en bit – des ordres.

En ce qui concerne les ordres à réponse immédiate, le nombre maximal d'ordres par seconde vaut :

$$\frac{STACK\_SIZE}{2 * TIMEOUT}$$

Sachant que $STACK\_SIZE = \min(MEM\_SPACE, 256)$; avec :

MEM_SPACE : le nombre maximal d'ordres pouvant être stockés simultanément en mémoire

256 correspond au nombre d'identifiants de conversation différents existant.

En ce qui concerne les ordres longs, STACK_SIZE correspond quasiment au nombre d'ordres pouvant être *effectués* simultanément (car 2*TIMEOUT est négligeable devant la durée d'exécution d'un ordre long). Le nombre maximal d'ordres par seconde dépend donc de la durée d'exécution de chaque ordre.

Il est souvent préférable de faire en sorte que le débit binaire soit le facteur limitant car il est alors plus simple de calculer la bande passante utilisée. Toutefois ce n'est pas toujours possible notamment à cause des ordres long dont l'exécution peut être arbitrairement longue. Il faut donc trouver un compromis.