Fonctionnement carte FPGA - E/S Asservissement
==========

Dip switches :
---
1 PIN27 : Frequence PWM, bit 1
2 PIN25 : Frequence PWM, bit 0
3 PIN24 : Predisiveur 1, bit 1
4 PIN23 : Prediviseur 1, bit 0
5 PIN22 : Prediviseur 0, bit 1
6 PIN21 : Prediviseur 0, bit 0
7 PIN18 : Sens cpt 3
8 PIN19 : Sens cpt 2
9 PIN16 : Sens cpt 1
10 PIN17 : Sens cpt 0

* Prédivision du sens :
1 , 2 , 4 , 16 corrspondant à
00, 01, 10, 11 sur les dip switches

Composition des trames sérielles

Toutes les trames sont en 8 bits, 1 stop, pas de parité, 115200 bauds

* Reception de la PWM :
4 trames par message : 2 trames pour moteur 1, puis 2 pour moteur 2 :
- '0' '0' F S 9 8 7 6 moteur 1
- '0' '1' 5 4 3 2 1 0 moteur 1
- '1' '0' F S 9 8 7 6 moteur 2
- '1' '1' 5 4 3 2 1 0 moteur 2

avec F : frein, S : sens, et PWM sur 10 bits numérotes de 9 à 0

* Emission des compteurs :
Emission échantillonnée de 10 octets consecutifs : (compteurs 16 bits)
0x55 0x55 MSB0 LSB0 MSB1 LSB1 MSB2 LSB2 MSB3 LSB3

RESET

RST PIN72

RS232

TX PIN58 (Point de vue DTE, donc RX pour le FPGA)
RX PIN60 (Point de vue DTE, donc TX pour le FPGA)
RTS PIN62

Sortie moteur 1

ENABLE PIN10
IN1 PIN8
IN2 PIN9

Sortie moteur 2

ENABLE PIN7
IN1 PIN5
IN2 PIN6

DIP-Switches
---
1 PIN27
2 PIN25
3 PIN24
4 PIN23
5 PIN22
6 PIN21
7 PIN18
8 PIN19
9 PIN16
10 PIN17

Entrées compteurs

Cpt 1-COD PIN35
Cpt 1-DIR PIN36

Cpt 2-COD PIN47
Cpt 2-DIR PIN48

Cpt 3-1 PIN51
Cpt 3-2 PIN52

Cpt 4-1 PIN53
Cpt 4-2 PIN54

Schéma E/S de la carte d'asservissement

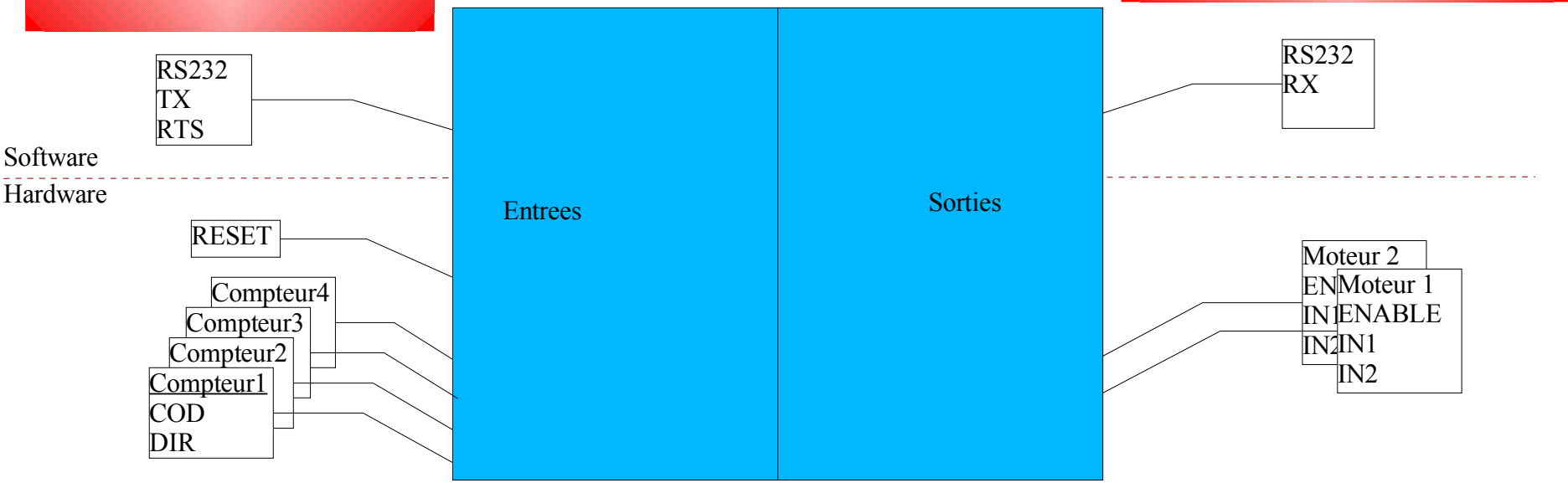
Message d'entrée :
4 trames de 8 bits :
- '0' '0' F S 9 8 7 6 moteur 1
- '0' '1' 5 4 3 2 1 0 moteur 1
- '1' '0' F S 9 8 7 6 moteur 2
- '1' '1' 5 4 3 2 1 0 moteur 2

F : frein, S : sens
PWM sur 10 bits numérotes de 9 à 0

Liaison série : 115200,8N1
(point de vue DTE ou HOST)

Message de sortie :
1 trame de 10 octets consecutifs
=> 4 compteurs de 16 bits :

0x55 0x55 MSB0 LSB0 MSB1 LSB1 MSB2 LSB2 MSB3 LSB3



Carte Asservissement

API carte Asservissement :

Input :

```
typedef struct {
        int frein;        // 1 ou 0
        int sens;         // 1 ou 0
        int puissance;  // sur 10 bits soit [0;1023]
} api_asserv_moteur_input;

typedef struct {
        struct api_asserv_moteur_input_t moteur1, moteur2;          // les donnees des 2 moteurs fournis par le soft a la carte
} api_asserv_input_t;
```

Output :

```
typedef struct {
        int valeur;       // sur 16 bits soit [0;65535]
} api_asserv_compteur_output_t;
typedef struct {
        struct api_asserv_compteur_output_t compteur1, compteur2, compteur3, compteur4;        // les 4 compteurs fournis par la carte au soft
} api_asserv_output_t;
```