PhilBob
février 2013

DOC dummy_12AX7

Etat logiciel

date	µC	remarques	PC	remarques
01/02	dummy_12AX7_V06	Modification du dictionnaire des commandes, abandon des test de sécurité du tube.		
28/01	dummy_12AX7_V05	Modification du format du vecteur de retour qui devient maintenant identique au format tdf.		
27/01	dummy_12AX7_V04	transmission en chaine de caractères sans les commentaires.	tx_rx_dummy_ 12AX7_V04	Dialogue avec le µC pour faire une exploration des paramètres
25/01	dummy_12AX7_V03	OK: transmission en chaine de caractères avec des commentaires.	Rien: utilisation avec cutecom.	
20/01	ping_pong_xx	Programmes simples pour mettre au point le type de transmission. Plusieurs versions		
14/01	dummy_12AX7_V02	Transmission en binaire: ne fonctionne pas chez Totof	tx_rx_dummy_ 12AX7_V01	Fonctionne chez moi

Au fil de l'eau

01/02/2013
Création de la V06.
Le dictionnaire des commandes est modifié. prog_µC ne gère plus la sécurité du tube. Cette tâche sera confiée au prog_PC.

28/01/2013
Création de la V05.

Modification de la V04 pour que le format de retour du vecteur de mesure soit au format tdf.

27/01/2013
Création de la version V04 pour le µC et le PC.
Corrections mineures par rapport à la V03.
Correspond à la version stable de la série dummy_12AX7 pour le µC.

25/01/2013
Après moult essais, le prog dummy_12AX7 fonctionne. Passage des transmissions en chaines de caractères

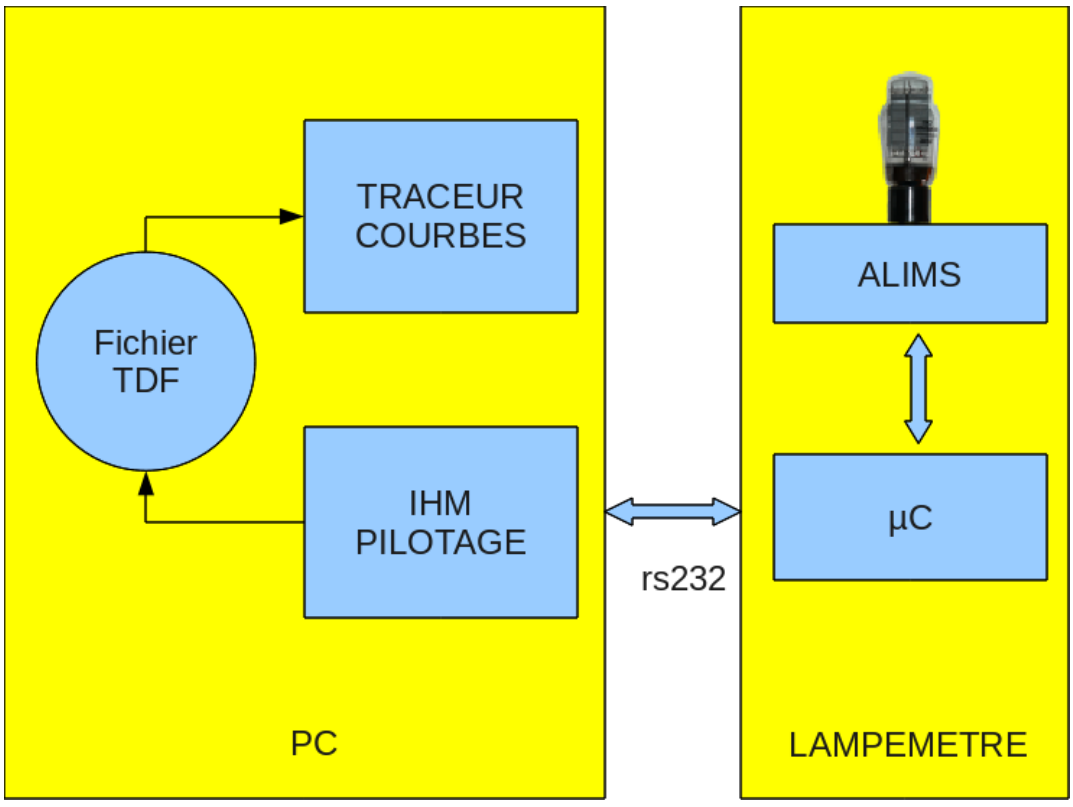
15/01/2013
Prog dummy_12AX7: ajout de la fonction STOP au dictionnaire. Cette fonction aura pour rôle, si le besoin s'en fait sentir, de mettre le montage en sécurité en coupant par exemple l'alim générale, (une sorte d'arrêt d'urgence)

Ajout d'une fonctionnalité d'attente (les leds de contrôle sont en mode chenillard en l'absence de commande). Ca fait un peu geek mais c'est pratique en phase de mise au point pour savoir si le µC attend une commande ou un paramètre supplémentaire.

14/01/2013
Début de rédaction de ce doc.

Introduction

Le projet global concerne la réalisation d'un lampemètre piloté par un PC. Le schéma de principe est indiqué ci dessous:



Ce doc décrit le programme prog_µC qui préfigure le soft final qui sera implanté dans un µC ATMega32 servant de pilote à un lampemètre. La mise au point de l'IHM de pilotage, du prog_µC et du protocole de dialogue, alors que l'électronique des alims n'existe pas encore, n'est pas directement réalisable.
Pour défricher cet aspect du projet, on réalise une maquette constituée par:
Coté PC un programme nommé tx_rx_dummy_12AX7_Vxx qui fait office de pilotage
Coté µC un programme nommé dummy_12AX7_Vxx qui fait office de prog_µC

Par commodité, le programme sur le PC est nommé prog_PC.
Par commodité, le programme sur le µC est nommé prog_µC.

Prog_PC ne comporte pas d'IHM mais présente toutes les fonctions de dialogue avec le matériel via une liaison rs232.

Prog_µC ne pilote pas d'électronique mais dispose de tous les blocs fonctionnels pour le faire. Ce programme retourne les grandeurs d'un modèle de triode 12AX7 afin d'être réaliste.

Principe de fonctionnement

Prog_µC est capable de simuler le réglage des différentes tensions du tube (Va,Vg1, Vg2, Vfil) puis de simuler la mesure de ces tensions ainsi que des courants associés (Ia, Ig1, Ig2).

Prog_µC est capable de simuler ces mesures à la demande de prog_PC. En réalité, les valeurs Va et Vg1 alimentent le modèle de triode qui calcule le courant de plaque en suivant une approximation de Norman Koren. Néanmoins, une bonne partie des mécanismes (routines) est en place et il suffira d'écrire le code de ces routines pour attaquer l'électronique.

De son coté, prog_PC est capable de faire une exploration paramétrique des grandeurs du tube sous test en pilotant Prog_µC, en récupérant les mesures simulées et en les stockant dans un fichier de résultat. Le rôle de prog_PC est de préfigurer les commandes qui seront transmises par le biais d'une IHM plus conviviale et de générer les fichiers nécessaires à d'autres applications (programme TCT et fichiers tdf).

Principe de communication

Pour des raisons pratiques, la communication s'appuie sur le protocole rs232. Le principe de communication est fondé sur la création d'un dictionnaire de commandes partagé par les deux programmes et un vecteur de données mesurées en retour.

Du PC vers le µC la trame transmise est:

Une chaine de caractères représentant le numéro de la commande suivi d'une chaine de caractères représentant la valeur associée à la commande. En retour, le µC renvoie une chaine de caractères représentant une indication sur l'interprétation de la commande.

Nota: Dans les toutes premières versions de prog_µC, la transmission des données était fondée sur des entiers et des flottants. La gestion des flux de données était alors très simple et rapide.

Malheureusement, il s'est avéré que, coté PC, tous les langages de programmation n'offrent pas la même souplesse que le C pour accéder à la représentation IEEE 754 des nombres. Comme le projet s'oriente vers un prog_PC écrit en Gambas, il a été décidé de transmettre des chaines de caractères
Pour la suite du document, on fera référence à des données type entier ou flottant mais le mécanisme de transmission s'effectue par chaines de caractères

Parmi les commandes, on distingue:

Les commandes informatives qui renseignent le µC, par exemple pour fixer Va max.
Les commandes exécutives qui enclenchent une action, par exemple pour régler Vfil,
Les commandes interrogatives qui demandent une mesure et donc provoquent une action combinée de réglage et de mesure.

Du µC vers le PC la trame transmise est:

Chaque commande reçue par le µC provoque l'envoi d'un code retour donnant une indication sur l'interprétation de la commande.

Dans le cas d'une demande de mesure, le µC renvoie un entier représentant une indication sur les dépassements éventuels suivi de six flottants (vecteur de données) représentant les mesures.

Dictionnaire des commandes

Commande émise par prog_PC	Code commande	Routine µC	Code retour du µC
sécurité			
SET_NUL	0	void fonc_void(float v);	99
SET_STOP	1	void arret(float v);	50
GET_VERS	2	void version(float v);	rien
Domaine des paramètres			
SET_VA_max	3	void set_VA_max(float v);	0
SET_VG2_max	4	void set_VG2_max(float v);	0
SET_VG1_max	5	void set_VG1_max(float v);	0
SET_VFIL_max	6	void set_VFIL_max(float v);	0
SET_VA_min	7	void set_VA_min(float v);	0
SET_VG2_min	8	void set_VG2_min(float v);	0
SET_VG1_min	9	void set_VG1_min(float v);	0
SET_VFIL_min	10	void set_VFIL_min(float v);	0

Réglage des tensions			
SET_VA	11	void set_VA(float v);	0, 1, 2
SET_VG2	12	void set_VG2(float v);	0, 1, 2
SET_VG1	13	void set_VG1(float v);	0, 1, 2
SET_VFIL	14	void set_VFIL(float v);	0, 1, 2
mesure			
MESURE	15	void mesure(float v);	0, 1

Signification des codes retour

Type de commande	Code de retour	signification
Sécurité		
SET_NUL	99	prog_PC a envoyé une commande absente du dictionnaire. Aucune action n'est entreprise mais le code 99 indique à prog_PC que la commande n'a pas réussi à être interprétée
SET_STOP	50	Demande d'arrêt d'urgence par prog_PC.
GET_VERS	rien	Renvoi une chaine de caractères indiquant la version logicielle et la date de création de cette version. Cette commande est surtout destinée à être activée via cutecom.
Domaine des paramètres	0	Commandes informatives
Réglage des tensions	0, 1, 2	0: la commande de tension est dans le domaine défini 1: la tension demandée est < à la tension min déclarée 2: la tension demandée est > à la tension max déclarée
mesure	0, 1	0: les tensions demandées sont dans le domaine 1: au moins une tension demandée est hors domaine des paramètres (voir vecteur retour)

Gestion de la sécurité

La gestion de la sécurité du tube n'est pas réalisée par prog_µC. Ce sera du ressort de prog_PC.

Néanmoins, prog_µC comporte des fonctions permettant de définir le domaine des paramètres mini maxi réalisables par le banc de test. Sont ainsi définies les grandeurs suivantes:

Va max, Va min	tension de plaque min et max réalisable par le banc de test.
Vg2 max, Vg2 min	tension d'écran min et max réalisable par le banc de test.
Vg1 max, Vg1 min	tension de grille min et max réalisable par le banc de test.
Vfil max, Vfil min	tension de filament min et max réalisable par le banc de test.

Après avoir défini ce domaine, toute commande de tension hors domaine ne sera pas exécutée et renverra un code retour indiquant la nature du dépassement.

Pour l'arrêt d'urgence, prog_µC est doté d'une fonction (commande 1) qui peut être programmée, par exemple, pour couper l'alimentation du montage. Cet arret d'urgence envoie 50 comme code retour.

Nota: La commande 0 est reliée à une fonction « nulle » dont le but est de ne rien faire si ce n'est que de renvoyer le code retour 99. Cette fonction est appelée lorsque prog_µC est activé avec un numéro de commande ne faisant pas partie de son dictionnaire. Le code retour indique au prog_PC que la commande n'a pas été comprise.

Vecteur retour

Lors d'une mesure réussie (code retour = 0), le vecteur retour est constitué de 6 grandeurs envoyées dans l'ordre suivant:

Va Ia Vg2 Ig2 Vg1 Ig1 (les valeurs sont séparée par un blanc)

Ces 6 grandeurs sont contenues dans une chaine de caractères se terminant par un LF (valeur dec=10). Le format du vecteur de retour est du type TDF (format développé par Yves et utilisable par son application TCT qui permet, entre autre, de tracer les courbes caractèristiques du tube sous test)

Si la mesure échoue (code retour !=0), tous les membres du vecteur sont nuls sauf ceux qui ont provoqué le dépassement. 1 signifiant limite basse atteinte, 2 signifiant limite haute atteinte.

Unités

Que ce soit de prog_PC vers prog_µC ou l'inverse, les unités employées sont:

grandeur	**unité**
Tension	Le Volt
Courant	Le milli Ampère

Passage de valeurs

Pour le numéro de commande, c'est un entier positif.
Pour une valeur, c'est un flottant.

type	min	max
Entier positif	0	255
flottant	-32765 avec 3 décimales	+32765 avec 3 décimales

A noter que l'arrondi décimal se fait par défaut, c'est à dire à la décimale inférieure.

Limitations:

Bien qu'étant doté d'un interpréteur de valeurs flottantes, prog_µC ne sait lire que les valeurs exprimées de manière simple. La notation exponentielle n'est pas reconnue.
Le séparateur décimal est le point « . »

Quelques exemples de valeurs transmises:

10
-255
10.99
-88.65
-0.754
-.145
.77

Quelques exemples de valeurs non interprétées:

+47 (le + n'est pas reconnu)
19.4e-5 (le terme e-45 n'est pas reconnu)
  12.63 (il y a un ou plusieurs « blancs » qui précèdent la valeur)
7,32 (la virgule n'est pas reconnue)