Eurogam project EDOC126

Protocoles between electronic card and software
tcard
protocole between test card and register server on station
protocole between test card and data base
protocole between test card and mini spectrum serveur

Edition : 1.0
Date : july 1991
Authors: B.HUMBERT, Ch.RING & G.ZEHNACKER
: MMA.ALEONARDC, C.DIARRA, P.KADIONIK

SATD
Centre de Recherches Nucléaires de Strasbourg
CENG Bordeaux
CNRS-IN2P3 France

1 Philosophie générale

Le logiciel test card crée des lignes de commandes (ou requêtes) pour différents serveurs:
- le register server
- le data base server
- le mini serveur de spectre dans le ”resource manager”
Il attend en retour un ou des messages .

Chacune de ces requêtes activera une tâche dans la station client qui préparera des infos pour la transmission sur le reseau ou en local. A travers une RPC la demande est transmise à la tâche serveur qui éxécutera l'action demandée (par ex: extraction de la data base, transmission d'une valeur de registre, ou extraction d'un spectre du ”resource manager”)

En retour, on renverra la ligne de la commande émettrice (”commande: xxx”), puis un status (”status: ok” ou ”status: autre”) indiquant la bonne execution ou non de la demande puis enfin eventuellement des informations en retour pour les commande de lecture d'un registre (”value: xxx”), de la data base (”value: name, value, unit , offset, gamme”) ou d'un spectre.

La ligne de commande en retour permet de connaire l'appartenance des lignes de status et de resultats qui suivent.

2 Protocole entre test card et ”register server”

2.1 syntaxe

read_vxi <card_name> <detector_number> <detector_register>
read_vxi <card_name> <detector_number> - <detector_number> <detector_register>
read_vxi <card_name> <detector_number> , <detector_number> , <detector_number> <detector_register>
write_vxi <card_name> <detector_number> <detector_register> <value> [<unit>]
write_vxi <card_name> <detector_number> , <detector_number> , <detector_number> <detector_register> <value> [<unit>]
write_vxi <card_name> <detector_number> - <detector_number> <detector_register> <value> [<unit>]
scope_vxi <card_name> <detector_number> <test_point_number> <type_of_signal>
button_vxi <card_name> <detector_number> <button_number> <status>

2.2 card_name

Voici les noms autorisés :

- trigger
- germanium
- bgo
- demon
- icare

2.3 detector_number

Le nombre de detecteur disponible sera trouvé à partir de la data base.

2.4 detector_register

Leur nombre depend de la carte ; nous obtiendrons leur nombre à partir de la "data base". Pour un detecteur germanium , il y en a actuellement 13.

ntsubsectiontest_point_number

Ces points se divise en plusieurs types. Pour l'instant les symboles utilisés sont L (logique) , A (analogique) , CF (contrôle de fenetre) et MT (mesure de tension). Au total pour le detecteur germasnium il y en a 17.

2.5 button_number

Ces objets auront 2 ètats possibles; pour le germanium , il y en aura 12.

2.6 exemples de commandes

read_vxi germanium 40 4
read_vxi germanium 40-50 4
read_vxi germanium 40,43,45 4
write_vxi germanium 40 4 250 ms
write_vxi germanium 40-50 4 250 ns
write_vxi germanium 40,43,45 4 250 ns
scope_vxi germanium 40 4 L2
scope_vxi germanium 40 1 A1
scope_vxi germanium 40 6 MT
scope_vxi germanium 40 7 CF
button_vxi germanium 40 4 ON
button_vxi germanium 40 4 OFF
button_vxi germanium 40 5 INT
button_vxi germanium 40 5 EXT
button_vxi germanium 40 6 LBU
button_vxi germanium 40 6 LEMOS

2.7 message en retour

2.7.1 read_vxi germanium

en retour on recevra
"commande: read_vxi germanium 40 1"
"status: ok" ou "status: OK" ou "status: xxxxx" puis
"value: 50 "

2.7.2 write_vxi germanium

en retour on recevra:
"commande: write_vxi germanium 40 1 100 mv"
"status: ok" ou "status: OK" ou "status : xxxx"

2.7.3 scope_vxi

en retour on recevra:
"commande: scope_vxi germanium 40 4 A1"
"status : ok" ou "status: OK" ou "status : xxxx"

2.7.4 button_vxi

en retour on recevra:
"commande: button_vxi germanium 40 4 on
"status : ok" ou "status: OK"ou "status : xxxx"

3 Protocole entre test card et "database"

3.1 syntaxe

read_database <detector_name> <detector_number> r<number>
read_database <detector_name> <detector_number> s<number>
read_database <detector_name> <detector_number> b<number>
write_database <detector_name> <detector_number> r<number> <value> [<unit>]

r symbole des registres
b symbole des boutons
s symbole des scopes

3.2 exemple

write_database germanium 40 r4 50 mv
read_database germanium 40 r4
read_database germanium 40 b10
read_database germanium 40 s10

3.3 messages en retour

3.3.1 read_database registre

commande transmise par le Test card : "read_database germanium 40 r1"
en retour on recevra
"commande: read_database germanium 40 r1"
"status: ok" ou "status: xxxxx" puis
"value: 1 seuil 50 mv 0 200"

3.3.2 read_database button

commande transmise par le Test card : "read_database germanium 40 b1"
en retour on recevra
"commande: read_database germanium 40 b1"
"status: ok" ou "status: xxxxx" puis
"value: 1 button1 on _ on off

3.3.3 read_database scope

commande transmise par le Test card : "read_database germanium 40 s15"
en retour on recevra
"commande: read_database germanium 40 s15"
"status: ok" ou "status: xxxxx" puis
"value: 1 scope1 A1 _ _ _

3.3.4 write_database registre

commande transmise par le Test card : "write_database germanium 40 r1 40 mv"
en retour on recevra:
"commande: write_database germanium 40 r1 40 mv"
"status: ok" ou "status : xxxx"

4 Protocole entre test_card et "Resource Manager"

IL s'agit de lancer ou d'arrêter la formation de spectres, puis de la visualiser.

4.1 syntaxe

Les actions prévues sont: select
start
stop
refresh
erase

Le détail du protocole est en cours d'étude.

4.2 fonctions

select : selectionne le type de spectre à constituer
start : demarre l'acquisition sur un ou des spectres
stop : arrete l'acquisition
refresh : rafraichit l'image des spectres
erase : efface les spectres dans le "Resource Manager"

5 Commandes transmises au "Resource Manager"

L'interface graphique va générer une commande qui sera transmise au "Ressource Manager". Cette commande en ASCII comportera une action à éxécuter par le "Resource Manager" et des arguments pour cette commande. Cette commande pourra aussi être entrée à partir de tout terminal ASCII ou de toute fenêtre terminal. Il sera même possible de faire un "script" enchainant les différentes commandes. Le transfert de cette chaine ASCII se fera naturellement à travers le rèseau par des RPC

5.1 fonctions de base

a) visualisation de signaux sur le scope
b) lecture de registre d'une carte VXI
c) écriture dans un registre d'une carte VXI
d) lecture "Slow ADC"
e) sélection d'un paramètre ADC d'une voie pour formation de spectre (16k) dans le "Resource Manager"
f) effacement d'un spectre dans le "Resource Manager"
g) démarrage de la formation de tous les histogrammes demandés
h) arrêt de la formation de tous les histogrammes demandés
i) lecture d'un spectre vers le logiciel "card_test" pour affichage.

5.2 Schéma de principe

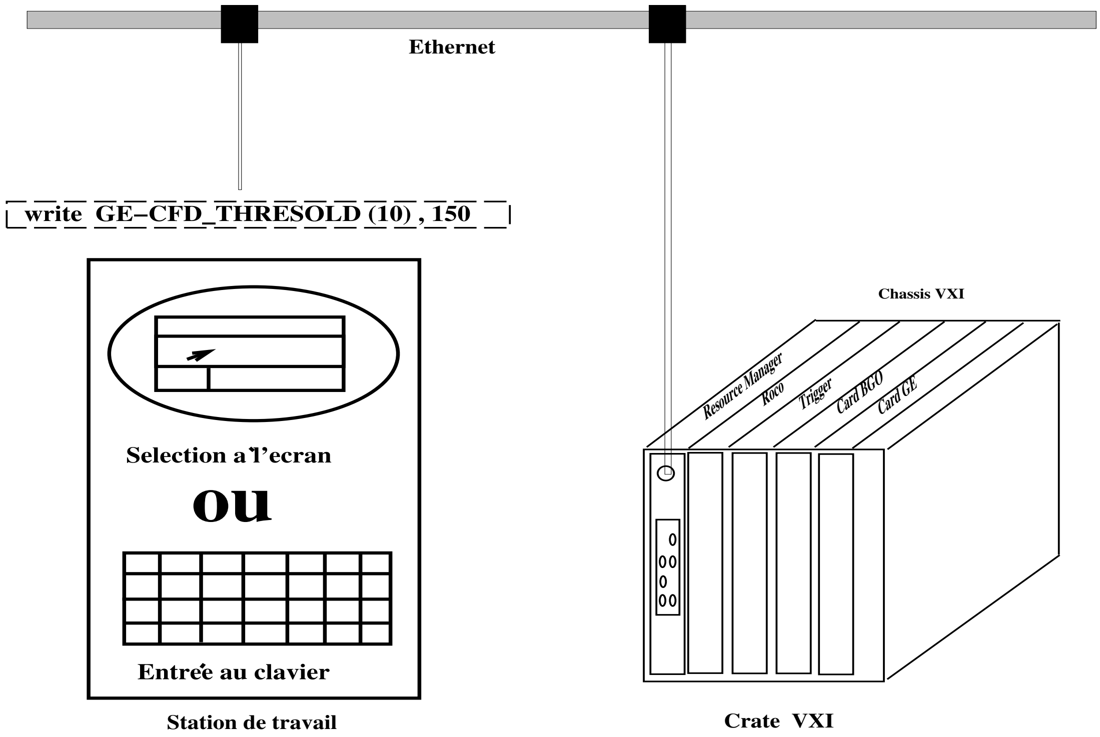

Fig 1 Commande "resource manager"

Dans la figure, write est le nom d'une fonction , GE_CFD_THRESOLD celui du nom du registre, 10 le numèro du détecteur et 150 la valeur à transmettre (ici sans unité précise).

5.3 Liste des commandes

a) Visualisation de signaux sur le "scope"

Nous pouvons estimer qu'il s'agit d'une écriture particulière. Ainsi la commande envoyée est par exemple:

" scope_vxi germanium 15 1 A1 " signifie envoyer la sortie 20Mev de l'amplificateur (numéroté 1) du détécteur Germanium (numéro 15) sur la première sortie analogique du "Resource Manager". Au niveau de la station de travail, le fichier "voies_crate" indique à quel serveur (c'est à dire dans quel chassis se trouve le voie) il faut transmettre la commande. Le "Resource Manager" qui execute celle-çi doit réaliser les actions suivantes.

1. **Pour toutes les cartes Germanium**
   - **a) Lecture de l'AMC (Analog Multiplexer Control) +**

b) **Modification de l'AMC : écriture de 0 dans l'octet concerné (octet MSB si A2, octet LSB si A1), l'autre octet doit conserver sa valeur (Mask)**

c) **Ecriture de l'AMC modifié**

2. **Pour toutes les cartes BGO**

   a) **lecture de l'AMC +**

   b) **Modification de l'AMC : écriture de 0 dans l'octet concerné (octet MSB si A2, octet LSB si A1), l'autre octet doit conserver sa valeur (Mask)**

   c) **Ecriture de l'AMC modifié**

3. **Pour la carte (résultant du numéro de la voie sélectionnée)**

   a) **Lecture de l'AMC +**

   b) **Modification de l'AMC : écriture du numéro de la voie (1$\leq$ channel Nb $\leq$ 6) dans les bits 2-0 de l'octet concerné (octet MSB si AL1, octet LSB si AL2) et du numéro de paramètre sélectionné (0 $\leq$ paramètre_NB $\leq$ 7) dans les bits 7-3 de l'octet concerné**

   c) **Ecriture de l'AMC modifié**

b) **Lecture de registre d'une carte VXI**

exemple :
"read_vxi germanium 70 2 " signifie lecture du seuil ("CFD THRESOLD") associée au Germanium 70. Au niveau de la station de travail, le fichier "voies_crate" indique à quel "Register Server" la commande est envoyée. Le fichier "voie_crate" permet de savoir dans quel chassis se trouve la voie concernée. Ainsi nous savons à quel "Register Server" de quel "Resource Manager" nous devons transmettre la commande .

Le "Resource Manager" qui execute cette commande doit réaliser les actions suivantes:

1. Rechercher l'adresse associée à CFD THRESOLD(7)
2. Effectuer une lecture VME à cette adresse
3. Transmettre le résultat à la station de travail appelante en acquittant l'action ("Retour RPC")

c) Ecriture dans un registre d'une carte VXI

exemple:
" write_vxi germanium 45 4 80 ns " signifie écriture de la largeur ("CFD WIDTH") du discriminateur à fraction constante ("CFD") de la voie germanium ("GE") 45. Cette largeur est de 80 nano seconde ("80 ns"). Après avoir déterminé le nom du "Server Register" à l'aide du fichier "voie_crate", le "Resource Manager"qui execute cette commande doit réaliser les actions suivantes:

1. Rechercher l'adresse associée à GE_CFD_WITDH(45)
2. Convertir les 80 NS en une valeur digitale
3. Effectuer l'écriture VME à l'adresse obtenue en 1.
4. Acquitter l'action ("Retour RPC").

d) Lecture du "SLOW ADC"

exemple :
" read_vxi germanium 2 SLOW_ADC " signifie lecture ("read_vxi") du convertisseur lent ("SLOW ADC") contenu dans le "Ressource Manager" du chassis VXI numéro 2. Ce "Resource Manager" pour effectuer cette lecture doit réaliser les actions suivantes:

1. Présenter l'adresse du Slow ADC (1) sur le bus externe P3 ("Address")
2. Envoyer l'ordre de codage (0) sur le bus externe P3 ("Data")
3. Lire le résultat de codage sur le bus externe P3 ("Data")
4. Répéter 1 2 et 3 N fois
5. Faire la moyennew sur les N mesures
6. Transmettre la moyenne obtenue à la station de travail appelante en acquittant l'action ("Retour RPC")

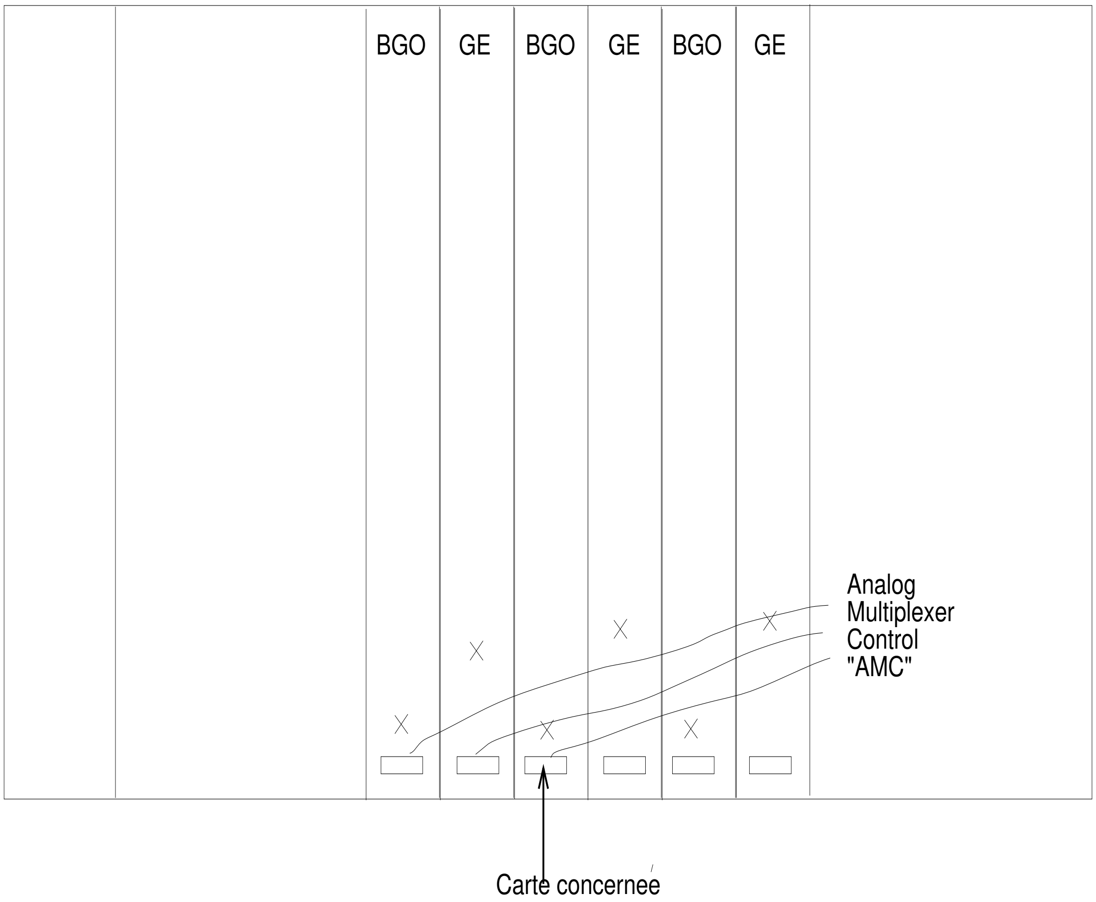

Figure 2 AMC lecture/ecriture