# Systeme informatique de commande du robot V 80

V. Dupourque

INRIA

CENTRE DE ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél. (3) 954 90 20

Rapports Techniques

N° 35

# SYSTÈME INFORMATIQUE DE COMMANDE DU ROBOT V 80

Vincent DUPOURQUE

Avril 1984

Abstract

This report describes the technical details of the hardware and software developed to control the 6 degrees of freedom manipulator V80 . The main points are :

- electrical adaptations (schemes)
- features of the Robot Operating System
- manipulator control and command algorithms (position control loop, trajectory generation.....)
- geometric transformations (direct and inverse)
- methodology of the software development (libraries, command files...)

A- VINCENT DUPOURQUE

Résumé

Ce document présente les aspects techniques de la mise en place du système informatique et de la commande de base du manipulateur ACMA V80 de l'INRIA.

C'est un document avant tout technique, dans lequel on retrouve tous les détails pratiques tels que :

- modifications de la baie de commande (schémas de câblage),
- caractéristiques du système d'exploitation (moniteur robot),
- commande minimum (asservissement, génération de trajectoire),
- modèle géométrique, direct et inverse,
- méthodologie du développement logiciel mise en place.

SOMMAIRE

b - Driver liaison calculateur

2.5.2. Handlers d'entrées/sorties

a - TRAP # 7 : entrées/sorties console opérateur

b - TRAP # 8 : entrées/sorties calculateur extérieur

2.5.3. TRAP #5 : utilitaires

2.6. Initialisation - Implantation mémoire

2.7. Utilitaires de conversion

2.8. Sous-programmes divers

3. Commande/Contrôle du manipulateur

3.1. Asservissement

3.2. Contrôle des déplacements

3.2.1. Déplacement en mode libre

3.2.2. Déplacement en ligne droite

3.3. Modèle géométrique

3.3.1. Formulation

3.3.2. Difficultés de mise en oeuvre

a - Singularités

b - Temps de calcul

c - Précision

d - Calcul d'erreurs - Estimation des incertitudes

3.3.3. Codage des variables

a - Codage des angles

b - Codage des longueurs

c - Codage des fonctions trigonométriques

d - Résumé des différents codages

3.3.4. Fonctions mathématiques

a - Interpolation linéaire

b - Optimisation de l'interpolation

c - Arctangente

1. Matériel.

Le manipulateur utilisé est un robot vertical 80 (V80) conçu par les services de recherches de la Régie Renault et fabriqué par la Société Acma Robotique.

Ses caractéristiques sont :

- commande hydraulique (pression de service : 110 bars)
- 6 degrés de liberté (rotations)
- vitesses maxi : 1 rad/s sur les 3 premiers axes
  3 rad/s sur les 3 axes du poignet
- une charge manipulable de 80 kg
- une précision de positionnement de 0.5 mm, avec une répétabilité de $\pm$ 0.2 mm.

Il a pour applications industrielles principales :

- le soudage par points
- la découpe
- la manutention de charges lourdes.

Il est commercialisé avec une baie de commande V3 équipée d'un calculateur Mitra (SEMS), ou V5 à base de microprocesseurs 8086 (INTEL).

Ce rapport à pour but de décrire le système à base du microprocesseur 68000 (Motorola) développé à l'INRIA et implanté dans une baie V3, en remplacement du Mitra d'origine.

1.1. Vue générale de la baie.

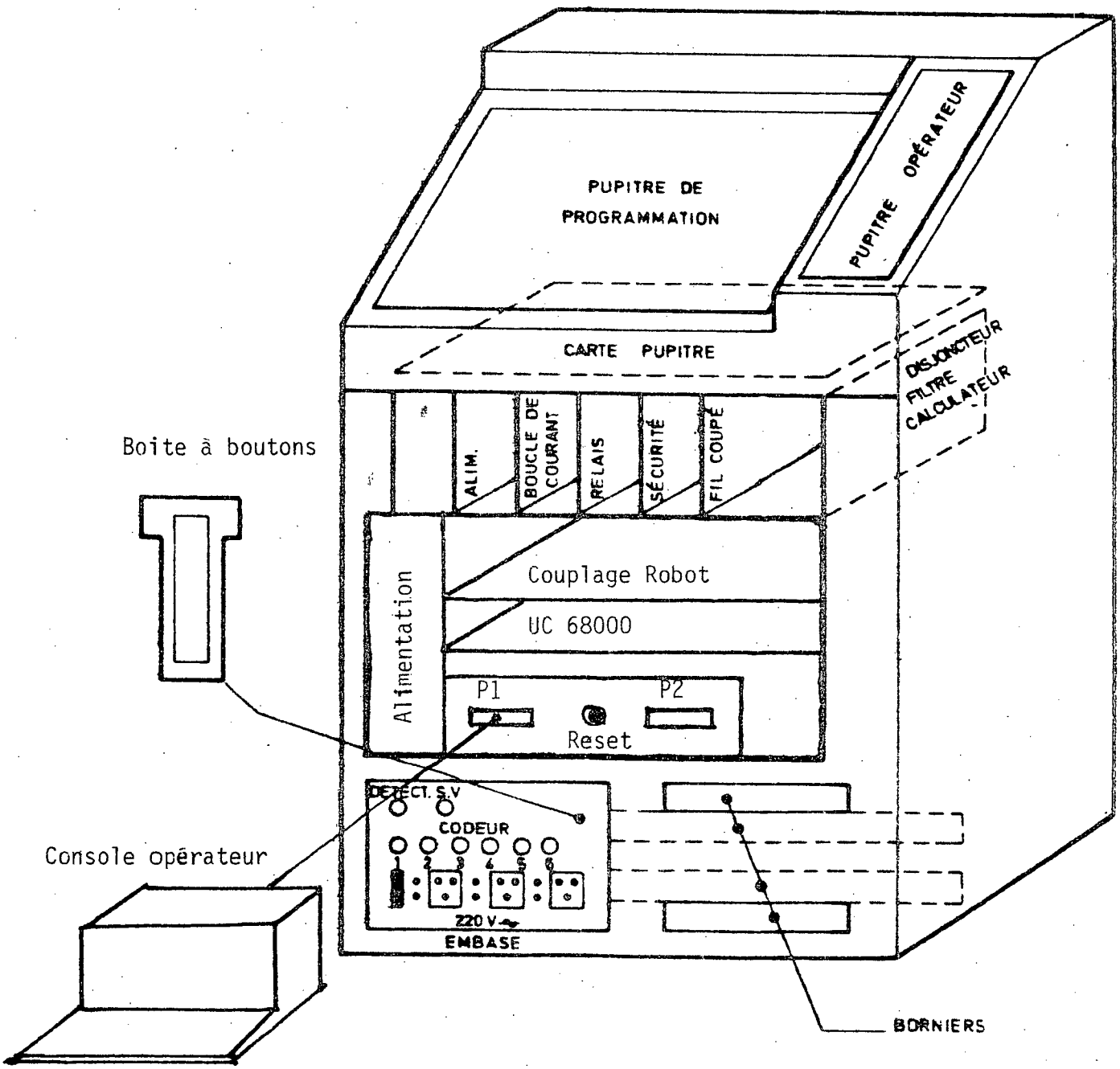



Fig. 1 : Vue générale de la baie de commande modifiée

Les détails de la partie constructeur sont donnés dans < 1 >.

Les modifications apportées à la baie d'origine ont été (Fig. 1) :

. Remplacement de l'ensemble calculateur Mitra-cartes d'interfaces par un fond de panier Versabus et 2 cartes : une unité centrale à base de 68000 et une carte d'interface dite carte de couplage robot.

. Mise en place d'une alimentation spécifique pour le Versabus, branchée sur le secteur et indépendante de la baie. Seules les masses sont communes.

. Montage de 2 liaisons séries accessibles dans la baie (P1 et P2).

. Sortie du reset microprocesseur.

1.2. Organisation fonctionnelle (fig. 2)

La carte de couplage robot assure l'interface entre le calculateur numérique et l'électronique analogique de puissance de la baie.

Les ordres provenant de la console opérateur (P1) ou d'un calculateur extérieur (P2) sont interprétés par l'unité centrale puis répercutés sur le manipulateur ou la baie par l'intermédiaire de la carte de couplage.

Actuellement, 2 emplacements sont encore disponibles dans le fond de panier Versabus pour étendre la puissance du calculateur.

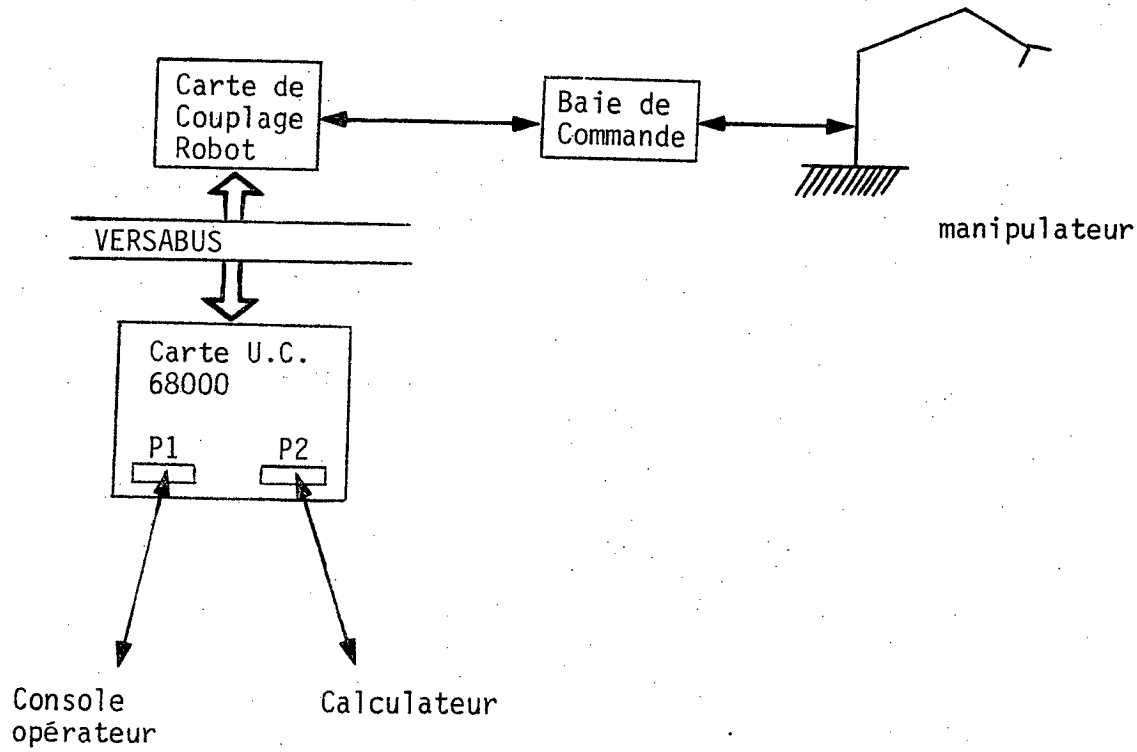



Fig. 2 : Organisation fonctionnelle de la commande du manipulateur V80

1.3. Carte de couplage robot (Fig. 3)

Développée à l'INRIA, elle assure le lien entre l'unité centrale et l'électronique de puissance. Ses caractéristiques sont :

- 8 mots mémoires ($FFFE10 à $FFFE1E)
- 8 compteurs ($FFFE00 à $FFFE0E) sur 16 bits
- 8 sorties analogiques ($FFFE20 à $FFFE2E)
- 16 entrées logiques isolées (optocoupleurs) ($FFFE80)
- 16 sorties logiques isolées (optocoupleurs) ($FFFE80)
- 32 registres non volatiles ($FFFE40 à $FFFE7F)
- 1 mot de validation de la procédure d'initialisation ($FFFE82)
- 1 octet d'état des compteurs ($FFFE42) pour détecter une rupture éventuelle de fils.

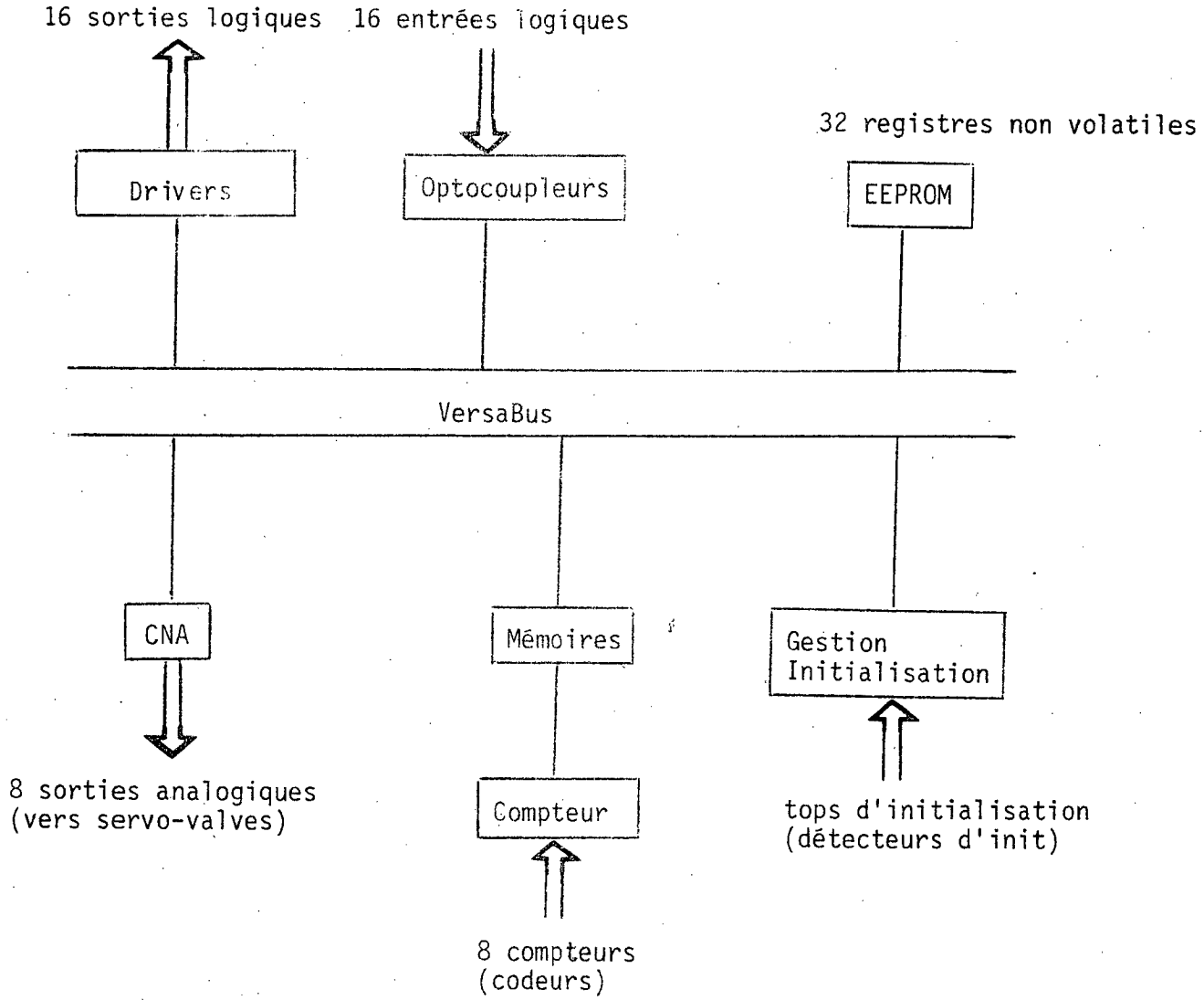



Fig. 3 : Synoptique de la carte de couplage robot

Les entrées logiques sont isolées électriquement par des optocoupleurs.

Les 8 codeurs incrémentaux sont multiplexés sur 1 seul compteur : les informations codeurs sont scrutées en permanence et la valeur liée à la position de l'axe incrémentée ou décrémentée selon le cas. La lecture des positions est faite par l'unité centrale via le Versabus, par Vols de cycles sur l'accès mémoire.

Les 8 sorties analogiques commandent les convertisseurs tension-courant placés sur la carte boucle de courant de la baie (commande des servo-valves).

Dans la version actuelle, 2 entrées compteurs et 2 sorties analogiques sont libres : elles pourront être affectées à la commande d'une pince proportionnelle.

Le module "gestion d'initialisation" assure, sous le contrôle de l'unité centrale, le recalage des compteurs.

Avant chaque arrêt du système, les valeurs des mémoires compteurs sont sauvegardées par programme dans des registres non volatiles (mémoire morte effaçable électriquement). A la remise sous tension, la position approximative du manipulateur est récupérée dans ces registres : dès lors, une initialisation automatique des compteurs est possible.

Cette initialisation consiste à passer par une position repérée à l'aide de proximètres. Dans la zone d'initialisation, et si le mot de validation de la procédure d'initialisation est actif, une interruption sera générée par le module d'initialisation, permettant ainsi la remise à zéro des mémoires compteurs dans une position physique toujours bien définie.

## 1.4. Schémas de câblage carte couplage robot - baie de commande

Les connecteurs J1 à J5 ont été montés en parallèle sur les connecteurs reliant le Mitra à la baie. Ces derniers n'ont, en principe, pas été modifiés.

Les tableaux suivants donnent les noms des signaux, leur adresse sur le connecteur de la carte couplage robot, leur nom et leur destination dans la baie ou sur un connecteur Mitra.

Les références sont les plans de la carte couplage ainsi que les schémas ACMA (folio, feuillet).

1.4.1. Liaisons codeurs incrémentaux

| Nom sur carte couplage | Connecteur | Nom sur connecteur robot | | Connecteur robot |
|---|---|---|---|---|
| Réf. : Schéma carte robot 1/3 Connecteurs J3 et J4 | | Réf. : 70500078 folio 32 Connecteur "COMPT" | | |
| Sa1 | J3-1 | S1 | | 5A - COMPT |
| $\overline{Sa1}$ | J3-20 | $\overline{S1}$ | | 6A - " |
| Sb1 | J3-2 | S2 | AXE 1 | 3A - " |
| $\overline{Sb1}$ | J3-21 | $\overline{S2}$ | | 4A - " |
| T1 | J3-3 | TOP-AXE 1 | | 31A - " |
| $\overline{T1}$ | J3-22 | $\overline{TOP-AXE\ 1}$ | | 30A - " |
| Sa2 | J3-4 | S1 | | 18A - COMPT |
| $\overline{Sa2}$ | J3-23 | $\overline{S1}$ | | 19A - " |
| Sb2 | J3-5 | S2 | AXE 2 | 16A - " |
| $\overline{Sb2}$ | J3-24 | $\overline{S2}$ | | 17A - " |
| T2 | J3-6 | TOP-AXE 2 | | 32A - " |
| $\overline{T2}$ | J3-25 | $\overline{TOP-AXE\ 2}$ | | 33A - " |
| Sa3 | J3-7 | S1 | | 27A - COMPT |
| $\overline{Sa3}$ | J3-26 | $\overline{S1}$ | | 28A - " |
| Sb3 | J3-8 | S2 | AXE 3 | 25A - " |
| $\overline{Sb3}$ | J3-27 | $\overline{S2}$ | | 26A - " |
| T3 | J3-9 | TOP-AXE 3 | | 36A - " |
| $\overline{T3}$ | J3-28 | $\overline{TOP-AXE\ 3}$ | | 37A - " |
| Sa4 | J3-10 | S1 | | 23A - COMPT |
| $\overline{Sa4}$ | J3-29 | $\overline{S1}$ | | 24A - " |
| Sb4 | J3-11 | S2 | AXE 4 | 20A - " |
| $\overline{Sb4}$ | J3-30 | $\overline{S2}$ | | 21A - " |
| T4 | J3-12 | TOP-AXE 4 | | 35A - " |
| $\overline{T4}$ | J3-31 | $\overline{TOP-AXE\ 4}$ | | 34A - " |
| Sa5 | J4-1 | S1 | | 13A - COMPT |
| $\overline{Sa5}$ | J4-20 | $\overline{S1}$ | | 14A - " |
| Sb5 | J4-2 | S2 | AXE 5 | 11A - " |
| $\overline{Sb5}$ | J4-21 | $\overline{S2}$ | | 12A - " |
| T5 | J4-3 | TOP-AXE 5 | | 40A - " |
| $\overline{T5}$ | J4-22 | $\overline{TOP-AXE\ 5}$ | | 41A - " |

| Nom sur carte couplage | Connecteur | Nom sur connecteur robot | Connecteur robot |
|---|---|---|---|
| Réf. : Schéma carte 1/3<br>Connecteurs J3 et J4 | | Réf. : 70500078<br>folio 32<br>Connecteur "COMPT" | |
| Sa6 | J4-4 | S1 ] | 9A - COMPT |
| $\overline{Sa6}$ | J4-23 | $\overline{S1}$ ] | 10A - " |
| Sb6 | J4-5 | S2 ] AXE 6 | 7A - " |
| $\overline{Sb6}$ | J4-24 | $\overline{S2}$ ] | 8A - " |
| T6 | J4-6 | TOP-AXE 6 | 39A - " |
| $\overline{T6}$ | J4-25 | $\overline{TOP\text{-}AXE\ 6}$ | 38A - " |
| | | Schéma (folio 13)<br>Réf. : Connecteur "COMPT"<br>folio 32 | |
| $\overline{INI}$ | J3-34 | INIT AXE 1 | 38B - COMPT |
| $\overline{IN2}$ | J3-35 | INIT AXE 2 | 34B - " |
| $\overline{IN3}$ | J3-36 | INIT AXE 3 | 30B - " |
| $\overline{IN4}$ | J3-37 | INIT AXE 4 | 41B - " |
| $\overline{IN5}$ | J4-34 | INIT AXE 5 | 33B - " |
| $\overline{IN6}$ | J4-35 | INIT AXE 6 | 37B - " |
| IN1 } | J3-15 | +24V AXE 1 → | Vers bornier Sécurité Robot |
| IN2 } | J3-16 | +24V AXE 2 | |
| IN3 } à relier | J3-17 | +24V AXE 3 | |
| IN4 } entre eux | J3-18 | +24V AXE 4 | |
| IN5 } | J4-15 | +24V AXE 5 | folio 5, 70500078 |
| IN6 } | J4-16 | +24V AXE 6 | |

Le principe du câblage des détecteurs d'initialisation est donné par la fig. 4.

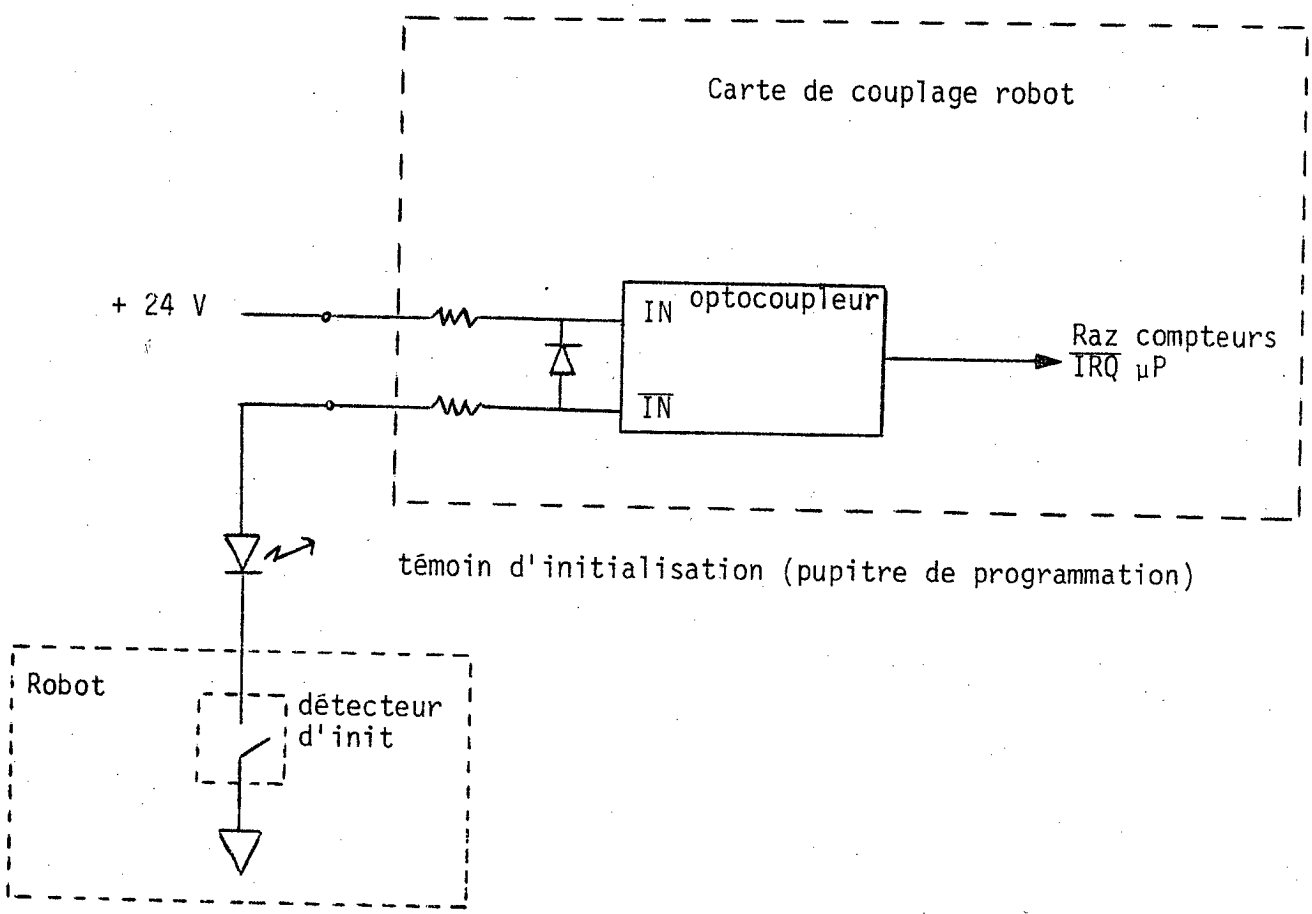



Fig. 4 : Câblage des détecteurs d'initialisation

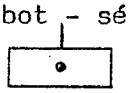
Le détecteur d'initialisation est un proximètre qui relie l'entrée $\overline{IN}$ de l'optocoupleur à la masse par l'intermédiaire d'une diode électroluminescente lorsqu'il est activé.

1.4.2. Sorties analogiques

| Nom sur carte coupleur Réf. : carte couplage robot 2/3 | Connecteur J1 | Nom sur connecteur Carte analogique Réf. : 7050078, folio 25 | Connecteur Carte analogique |
|---|---|---|---|
| SA1 | J1-1 | Consigne I1 | 9A |
| CZ1 | J1-15 | → | 1A (ou 1B) |
| SA2 | J1-3 | Consigne I2 | 10A |
| CZ2 | J1-2 | → | 1A (ou 1B) |
| SA3 | J1-4 | Consigne I3 | 11A |
| CZ3 | J1-18 | → | 1A (ou 1B) |
| SA4 | J1-6 | Consigne I4 | 12A |
| CZ4 | J1-5 | → | 1A (ou 1B) |
| SA5 | J1-7 | Consigne I5 | 13A |
| CZ5 | J1-21 | → | 1A (ou 1B) |
| SA6 | J1-9 | Consigne I6 | 14A |
| CZ6 | J1-8 | → | 1A (ou 1B) |
| SA7 | J1-10 | Consigne I7 | 38A |
| CZ7 | J1-24 | → | 1A (ou 1B) |
| SA8 | J1-12 | Consigne I8 | 39A |
| CZ8 | J1-11 | → | 1A (ou 1B) |
| | + sur le connecteur complémentaire | P1 10B → 19B<br>P1 12B → 20B | |
| | Connecteur F | Connecteur "CONS" GH 70500078, folio 34 | |
| | J1B-1 | | 27B |
| | J1B-3 | | 26B |
| | J1B-4 | | 25B |
| | J1B-6 | | 24B |
| | J1B-7 | | 23B |
| | J1B-9 | | 21B |

Dans le câble, tous les $SA_i$ sont torsadés avec les $CZ_i$.

1.4.3. Entrées logiques

| Nom sur carte couplage | Connecteur carte | Nom sur connecteur robot | Connecteur robot |
|---|---|---|---|
| Réf. : Schéma carte robot 2/3 connecteur J5 | | Réf. : manipulateur 70500078, folio 13 connecteurs E1, E2 et COMPT (32) | |
| EU1 | J5-1 | +24 V ⟶ | Vers bornier Robot - sécu  (folio 70500078, 5) |
| EU2 | J5-2 | | |
| EU3 | J5-3 | | |
| EU4 | J5-4 | | |
| EU5 | J5-5 | | |
| EU6 | J5-6 | | |
| EU7 (réserve) | J5-7 | | |
| EU8 | | | |
| $\overline{EU1}$ | J5-20 | NAXE 1 | D101 (pupitre) |
| $\overline{EU2}$ | J5-21 | NAXE 2 | D102 " |
| $\overline{EU3}$ | J5-22 | NAXE 4 | D103 " |
| $\overline{EU4}$ | J5-23 | SENS | A68 " |
| $\overline{EU5}$ | J5-24 | VIT | A69 " |
| $\overline{EU6}$ | J5-25 | PA (folio 15) | A95 " |
| $\overline{EU7}$ | J5-26 | PP " | A96 " |
| $\overline{EU8}$ | J5-27 | PS " | A97 " |
| EU9 | J5-9 | fin de course pince | bornier (f70500078, 5) |
| $\overline{EU9}$ | J5-28 | + 24V | bornier |
| EU10 | J5-10 | + 24V | " |
| $\overline{EU10}$ | J5-29 | SC12 (SEC1) | bornier (f70500078, 8) |
| EU11 | J5-11 | + 24V | " |
| $\overline{EU11}$ | J5-30 | SC22 (SEC2) | " |
| EU12 | J5-12 | + 24V | " |
| $\overline{EU12}$ | J5-31 | MAS2 (MASS1) | bornier (f70500078, 7) |

Il reste 3 entrées logiques disponibles.

Le mot d'entrée logique est :

| N°EL | N°bit | Fonction |
|---|---|---|
| 1 | 0 | NAXE 1 (boîte à boutons) |
| 2 | 1 | NAXE 2 " |
| 3 | 2 | NAXE 4 " |
| 4 | 3 | SENS " |
| 5 | 4 | VIT " |
| 6 | 5 | PA " |
| 7 | 6 | PP " |
| 8 | 7 | PS " |
| 9 | 8 | Fin course (pince) |
| 10 | 9 | Sécurité niveau 1 (SEC1) |
| 11 | 10 | Sécurité niveau 2 (SEC2) |
| 12 | 11 | Mise sous asservissement (MASS1) |
| 13 | 12 | Libre |
| 14 | 13 | Libre |
| 15 | 14 | Libre |
| 16 | 15 | Mode de fonctionnement (normal = 0, émulation = 1) |

## 1.4.4. Sorties logiques

| Nom sur la carte couplage | Connecteur carte | Nom sur connecteur robot | Connecteur robot |
|---|---|---|---|
| Réf. : carte couplage robot 2/3 connecteur J2 | | Réf. : connecteur SUEF 70500078, folio 16 | → idem |
| SU1 | J2-1 | CDG (chien de garde) | D43 (pupitre) |
| SU2 | J2-2 | SEC (sécurité soft) | D45 " |
| SU3 | J2-3 | VSAP (visualisation commande manuelle) | D10 " |
| SU6 | J2-6 | inutilisé | |
| SU7 | J2-7 | " | |
| SU8 | J2-8 | SUORD(f70500078, 17) Réf. Connecteur SUGH 70500078, folio 31 et 18 | D12 " |
| SU9 | J2-9 | VORD0 D00 | D60 " |
| SU10 | J2-10 | VORD1 D62 | D62 " |
| SU11 | J2-11 | VORD2 D64 | D64 " |
| SU12 | J2-12 | VORD3 D66 | D66 " |
| SU13 | J2-13 | VORD4 D68 | D69 " |
| SU14 | J2-14 | VORD5 D70 | D71 " |
| SU15 | J2-15 | VORD6 D72 | D73 " |
| SU16 | J2-16 | VORD7 D74 | D75 " |
| Masses : | forte section | | |
| retour masse sortie logique 1 | J2-20 | → noeud des masses : barre de zéro (folio 5, 70500078) | |
| " 2 | J2-21 | | |
| " 3 | J2-22 | | |
| " 6 | J2-25 | | |
| " 7 | J2-26 | | |
| " 8 | J2-27 | | |
| " 9 | J2-28 | | |
| " 10 | J2-29 | | |
| " 11 | J2-30 | | |
| " 12 | J2-31 | | |
| " 13 | J2-32 | | |
| " 14 | J2-33 | | |
| " 15 | J2-34 | | |
| " 16 | J2-35 | | |

Il reste 4 sorties logiques inutilisées, dont 2 câblées (6 et 7).

Le mot de sortie logique est :

| N°SL | N°bit | Fonction |
|---|---|---|
| 1 | 0 | CDG (chien de garde) |
| 2 | 1 | SEC (sécurité soft) |
| 3 | 2 | VSAP (visualisation boîte à boutons) |
| 4 | 3 | Libre |
| 5 | 4 | Libre |
| 6 | 5 | Libre ] câblés |
| 7 | 6 | Libre ] |
| 8 | 7 | BOB B (validation SLORD) |
| 9 | 8 | VORD0 (marche) |
| 10 | 9 | VORD1 (arrêt) |
| 11 | 10 | VORD2 (marche centrale) |
| 12 | 11 | VORD3 (marche asservissement) |
| 13 | 12 | VORD4 (raz sécu) |
| 14 | 13 | VORD5 |
| 15 | 14 | VORD6 (ouverture pince) |
| 16 | 15 | VORD7 (fermeture pince) |

Le relais VORD5 n'est affecté à aucune commande.

## 1.5. Carte KVM 68000

C'est une carte "monoboard" à base du microprocesseur 68000 Motorola, au format Versabus.

Ses principales caractéristiques sont :

- microprocesseur 68000 (16 bits), version 8 MHz
- 64 Kϕ de RAM dynamique ($8 à $FFFF)
- 32 Kϕ d'EPROM ($F00000 à $F07FFF)
- 2 ports d'E/S série programmés à 9600 bauds
- 32 lignes d'E/S parallèles
- 7 niveaux d'interruptions.

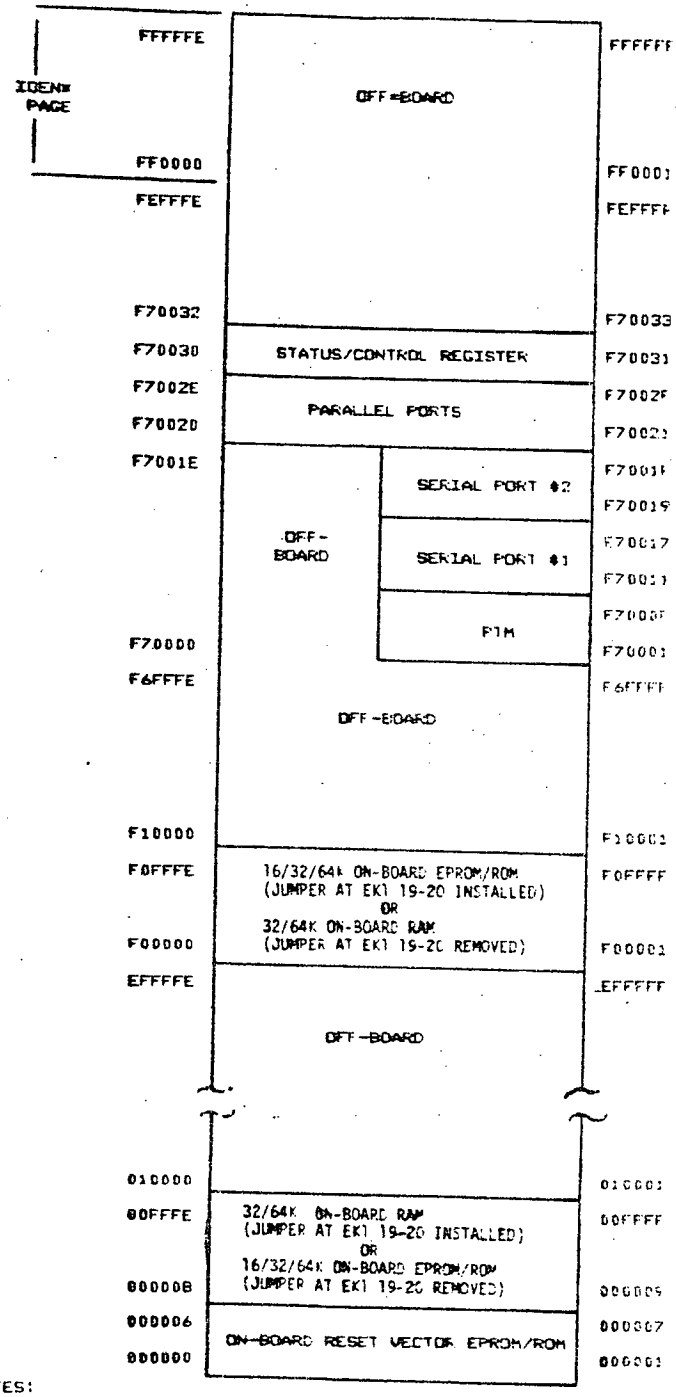
Elle est équipée du moniteur Versabug, contenant les utilitaires nécessaires à la mise au point de programmes simples.

Tous les détails concernant cette carte se trouvent dans :

"M68KVM01-2 Monoboard Microcomputer User's Guide"

La cartographie mémoire est donnée par la fig. 5 et le brochage des différents connecteurs par la fig. 6.

| DEVICE | ADDRESS RANGE | MODE | COMMENTS |
|---|---|---|---|
| STATUS REGISTER | F70030 | READ | |
| CONTROL REG. | F70030 | WRITE | |
| PARALLEL PORT#2 | F70027 | READ | LOWER BYTE STATUS REG. |
| PARALLEL PORT#2 | F70027 | WRITE | LOWER BYTE CONTROL REG |
| PARALLEL PORT#2 | F70026 | READ | UPPER BYTE STATUS REG. |
| PARALLEL PORT#2 | F70026 | WRITE | UPPER BYTE CONTROL REG |
| PARALLEL PORT#1 | F70025 | READ | LOWER BYTE STATUS REG. |
| PARALLEL PORT#1 | F70025 | WRITE | LOWER BYTE CONTROL REG |
| PARALLEL PORT#1 | F70024 | READ | UPPER BYTE STATUS REG. |
| PARALLEL PORT#1 | F70024 | WRITE | UPPER BYTE CONTROL REG |
| PARALLEL PORT#2 | F70023 | R/W | LOWER BYTE PERIPHERAL DATA REGISTER |
| PARALLEL PORT#2 | F70022 | R/W | UPPER BYTE PERIPHERAL DATA REGISTER |
| PARALLEL PORT#1 | F70021 | R/W | LOWER BYTE PERIPHERAL DATA REGISTER |
| PARALLEL PORT#1 | F70020 | R/W | UPPER BYTE PERIPHERAL DATA REGISTER |
| SERIAL PORT #2 | F7001F | READ | READ COMMAND REGISTER |
| SERIAL PORT #2 | F7001F | WRITE | WRITE COMMAND REGISTER |
| SERIAL PORT #2 | F7001D | READ | READ MODE REGISTERS |
| SERIAL PORT #2 | F7001D | WRITE | WRITE MODE REGSISTER |
| SERIAL PORT #2 | F7001B | READ | STATUS REGISTER |
| SERIAL PORT #2 | F7001B | WRITE | SYN1/SYN2/DLE REGISTER |
| SERIAL PORT #2 | F70019 | READ | RECEIVE DATA REGISTER |
| SERIAL PORT #2 | F70019 | WRITE | TRANSMIT DATA REGISTER |
| SERIAL PORT #1 | F70017 | READ | READ COMMAND REGISTER |
| SERIAL PORT #1 | F70017 | WRITE | WRITE COMMAND REGISTER |
| SERIAL PORT #1 | F70015 | READ | READ MODE REGISTERS |
| SERIAL PORT #1 | F70015 | WRITE | WRITE MODE REGSISTER |
| SERIAL PORT #1 | F70013 | READ | STATUS REGISTER |
| SERIAL PORT #1 | F70013 | WRITE | SYN1/SYN2/DLE REGISTER |
| SERIAL PORT #1 | F70011 | READ | RECEIVE DATA REGISTER |
| SERIAL PORT #1 | F70011 | WRITE | TRANSMIT DATA REGISTER |
| PTM | F7000F | READ | READ LSB BUFFER REG. |
| PTM | F7000F | WRITE | WRITE TIMER#3 LATCHES |
| PTM | F7000D | READ | READ TIMER#3 COUNTER |
| PTM | F7000D | WRITE | WRITE MSB BUFFER REG. |
| PTM | F7000B | READ | READ LSB BUFFER REG. |
| PTM | F7000B | WRITE | WRITE TIMER#2 LATCHES |
| PTM | F70009 | READ | READ TIMER#2 COUNTER |
| PTM | F70009 | WRITE | WRITE MSB BUFFER REG. |
| PTM | F70007 | READ | READ LSB BUFFER REG. |
| PTM | F70007 | WRITE | WRITE TIMER#1 LATCHES |
| PTM | F70005 | READ | READ TIMER#1 COUNTER |
| PTM | F70005 | WRITE | WRITE MSB BUFFER REG. |
| PTM | F70003 | READ | READ STATUS REGISTER |
| PTM | F70003 | WRITE | WRITE CONTROL REG.#2 |
| PTM | F70001 | READ | NO OPERATION |
| PTM | F70001(CR20=0) | WRITE | WRITE CONTROL REG.#3 |
| PTM | F70001(CR20=1) | WRITE | WRITE CONTROL REG.#1 |
| RAM(NOTE 2,3) | 000008-00FFFF | R/W | THE TWO BLOCKS SHOWN REPRESENT ALTERNATE PLACEMENTS FOR THE RAM (JUMPER SELECTABLE) |
| EPROM/ROM (NOTE 2,4) | F00000-F0FFFF | READ | 1ST 8 BYTES REDUNDANT WITH MEM.LOC. 000000-000007 |
| EPROM/ROM (NOTE 5) | 000000-000007 | READ | POWER-ON RESET + STACK POINTER |




NOTES:
1. JUMPER AT EK1(19-20) REMOVED.
2. JUMPER AT EK1(19-20) INSTALLED.
3. EXACT ENDING ADDRESS (XX7FFF OR XXFFFF) DEPENDENT UPON RAM OPTION (32K OR 64K BYTES). MAP DECODER RESERVES ENTIRE 64K BYTE BLOCK FOR FUTURE UPGRADE COMPATIBILITY.
4. EXACT ENDING ADDRESS DEPENDENT UPON EPROM/ROM DENSITY AND AND NUMBER INSTALLED. MAP DECODER RESERVES ENTIRE 64K BYTE BLOCK FOR FUTURE UPGRADE COMPATIBILITY.
5. EPROM/ROM PERMANENTLY ASSIGNED TO 1ST 8 BYTES FOR PROGRAM COUNTER AND STACK POINTER VECTORS.

Fig. 5 : Cartographie mémoire de la carte unité centrale

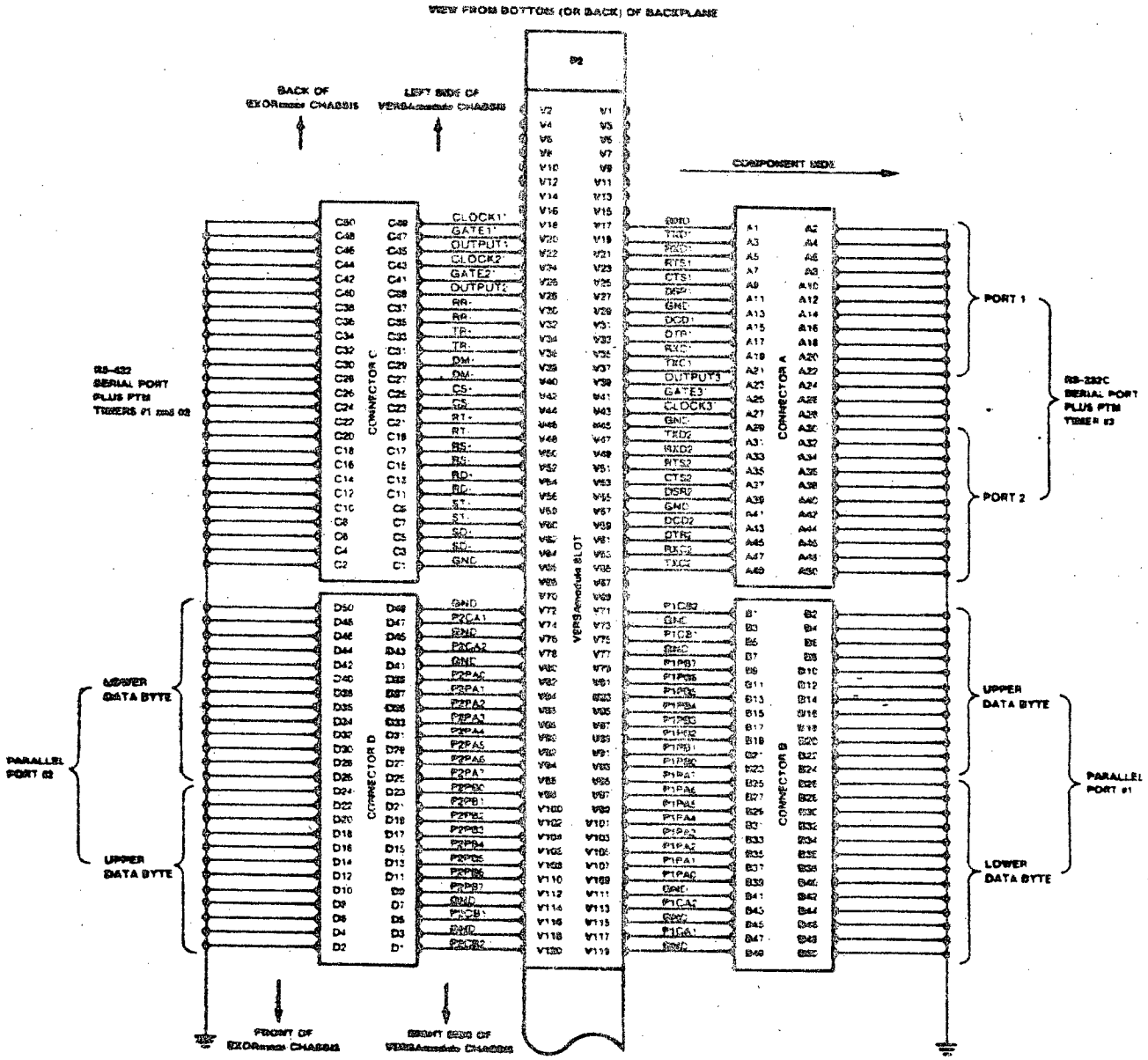



Fig. 6 : Brochage des prises de série et parallèle du Versabus

## 1.6. Alimentations

Le Versabus dispose d'une alimentation séparée (+5, +12, -12V). Seules les masses doivent être reliées à celles de la baie.

Les différentes masses sont :

terre (armoire)

0 volt puissance (24V)

0 volt analogique (servo-valves)

0 volt logique (calculateur)

La masse analogique est reliée au noeud des masses par l'intermédiaire de la broche P4-75 (ou 76, 77, 78, 79, 80) au noeud des masses de la baie, alors que le 0 V logique est pris directement sur l'alimentation séparée.

## 1.7. Modes de fonctionnement : normal ou émulation

2 modes de fonctionnement sont prévus pour la baie. En mode normal, toute intervention sur la baie doit être manuelle. En mode émulation, c'est le calculateur qui peut déclencher des actions telles que mise sous tension, mise en route de la centrale...

Pour cela, tous les interrupteurs importants du pupitre opérateur ont été doublés par un relais (de la carte relais).

Le mode de fonctionnement est sélectionné par un interrupteur sur le pupitre de programmation. Des modifications ont du être apportées à la

baie pour mettre en place ces modes.

Pour éviter tout problème à la mise sous tension, une double sécurité a été mise en place :

- ouverture de tous les relais par un reset
- validation de la carte relais par une sortie logique (SLORD valide la carte relais).

Le schéma adopté est donné par la fig. 7.

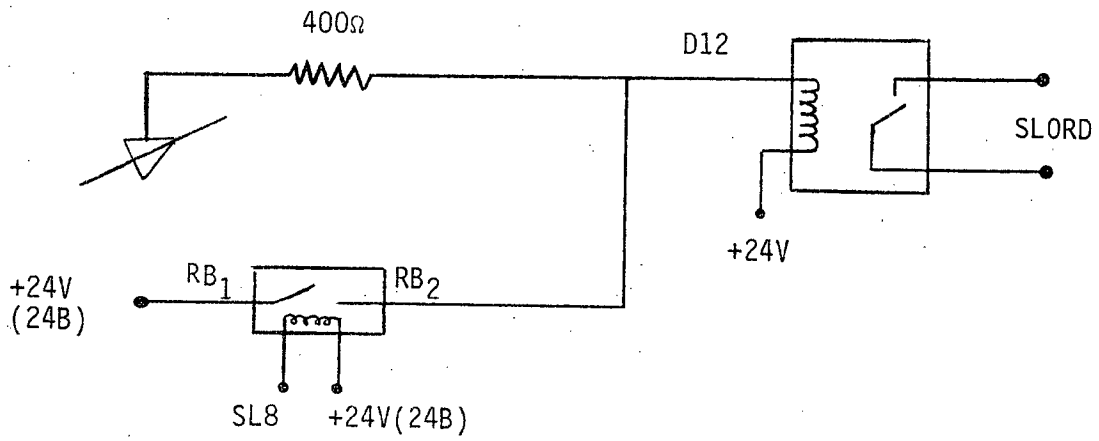



Fig. 7 : Chaîne de sécurité lors de la mise sous tension automatique de la baie

Tant que la sortie logique 8 n'est pas à 0, aucune information n'est prise en compte par la carte relais.

Les autres câblages nécessaires sont (réf. F70500078 7, 8 et 9) :

MISE SOUS TENSION : Arrêt 2 et 3 (bornier) sur D00 et D0 (bornier)
Marche 3 et 4 ( " ) sur D10 et D1 ( " )

MARCHE CENTRALE : 4H et 5H sur D20 et D2

MISE SOUS ASSERVISSEMENT : S17 - ⏚ sur D30 et D3

RAZ SECU : T20 - ⏚ sur D40 et D4

T20 se trouvant sous le pupitre de commande.

Le câblage de l'interrupteur de sélection de mode (sur le pupitre de programmation est donné fig. 8.

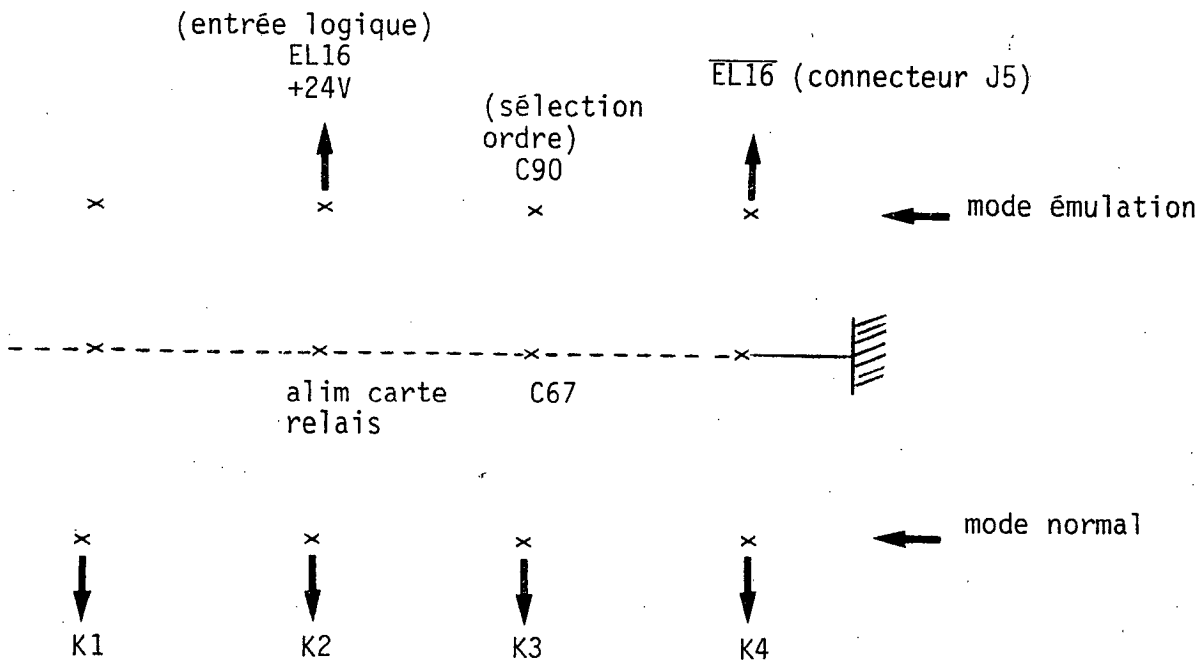



Fig. 8 : Câblage de l'interrupteur de sélection de mode (vu de dessous)

Avec : K1 = strap hors tension (réf. F70500078, 7)

K2 = alimentation carte relais (24B sur 24B1, réf. F70500078, 11)

K3 = validation des ordres envoyés par la carte relais (dévérouillage ordre, f70500078, 18)

K4 = mise à 0V $\overline{EU16}$ (si mode émulation)

## 1.8. Modifications sur la carte fils coupés

Le traitement d'une rupture de fils est réalisé par programme. Dès qu'une rupture est détectée, la carte de couplage robot le signale. La carte détection de fils coupés de la baie ne sert plus qu'à la visualisation.

On la neutralise en bloquant le relais FC (ref. F70500079, 9) : la piste A15 est coupée et forcée à la masse.

## 1.9. Câblage de la commande pince

La mise en place de la pince a nécessité les câblages suivants :

- dans le boîtier intermédiaire (boîtier bleu de raccordement électrique sur le bras 3 du manipulateur)

  A : commande distributeur hydraulique 1

  B : commande " " 2

  C : ——▷ distributeurs hydrauliques

  D : ——▷ détecteur fin de course (bleu)

  E : sortie détecteur fin de course (noir)

  F : entrée " " " (brun)

- dans le boîtier potence

  C + D ➡ —▷

- principe du branchement du détecteur de fin de course (fig. 9).

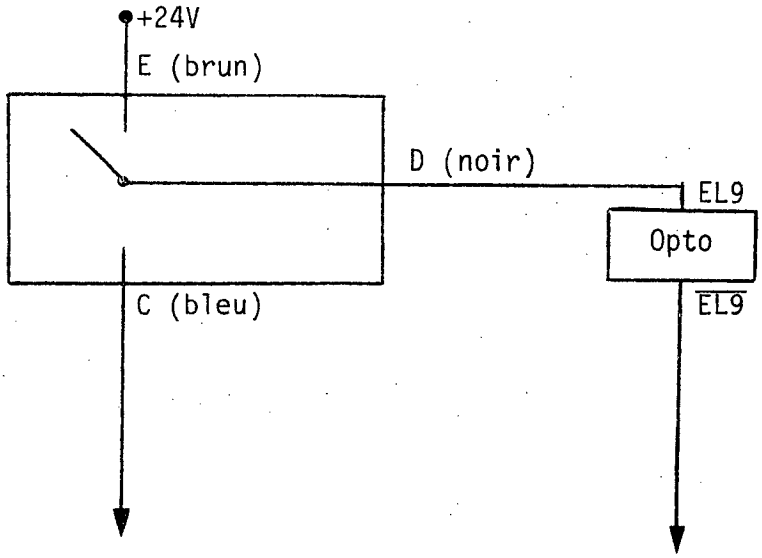



Fig. 9 : Branchement du détecteur de fin de course pince

. la commande d'ouverture/fermeture pince entraînant d'importants parasites, des varistances ont été soudées directement sur la carte relais (entre la sortie des relais 6 et 7 et la masse, réf. F70500079, 8).

1.10. Chaîne cinématique de la tête 3 axes (fig. 10 et 11)

Elle définit les couplages à prendre en compte pour la commande séparée des 3 derniers axes.

On appelle $m_i$ les consignes moteurs et $q_i$ les angles mécaniques.

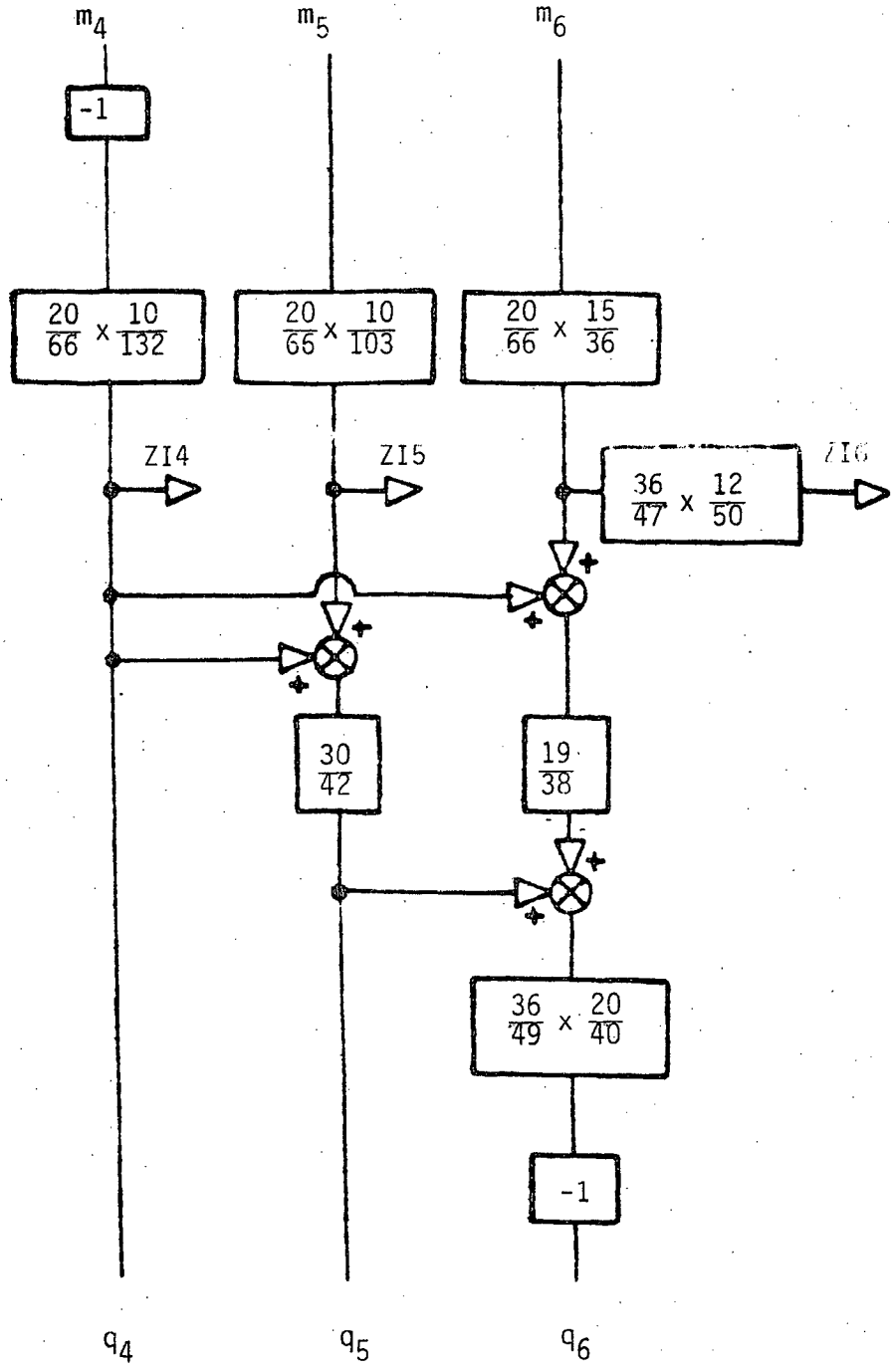



Fig. 10 : Chaîne cinématique directe de la tête 3 axes

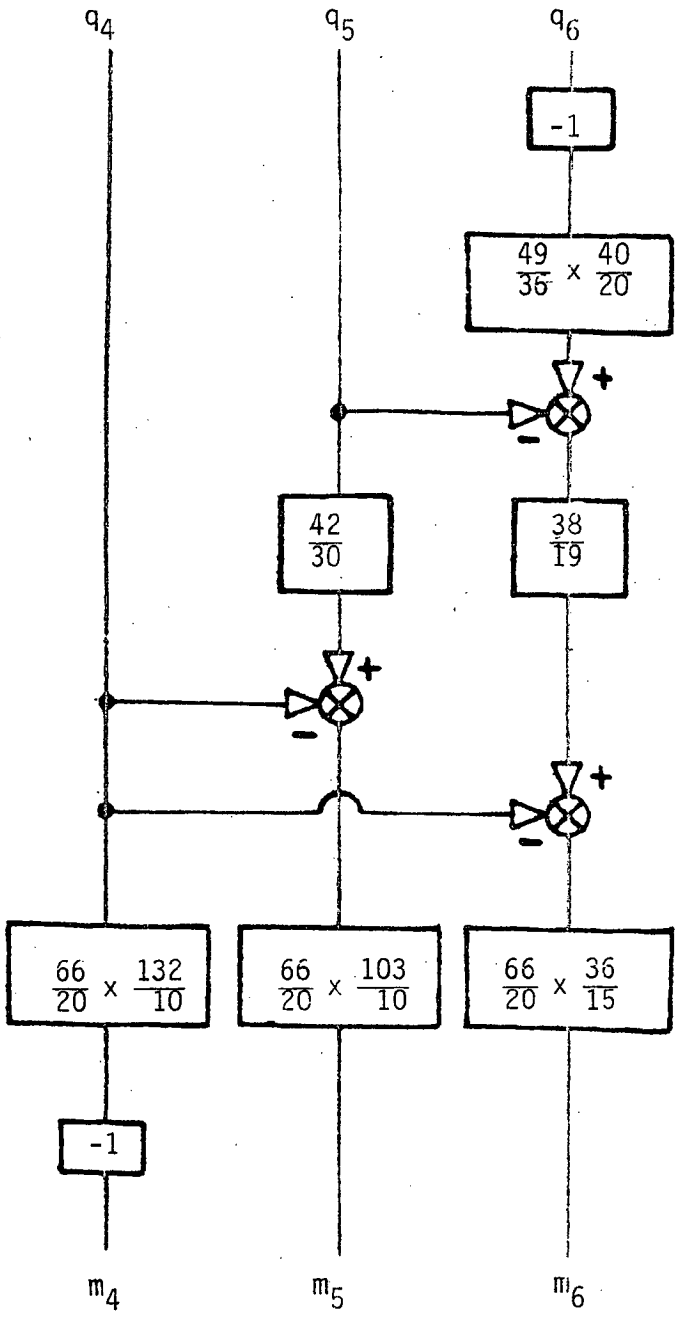



Fig. 11 : Chaîne cinématique inverse de la tête 3 axes

# VERTICAL I — P = 110 BARS — BANDE PASSANTE

| AXE | ROTATION OU TRANSL. | MOTEURS (DENZLER) TYPE | CYLINDRÉE $C_y$ cm³/rd | CONSTANTE DE COUPLE $C_9$ cm.daN/A | ½ VOLUME MORT $V_0$ cm³ | C MAX. À ω = 0 $P_{IC9}$ m.N | COEFF FUITES MOTEUR $K_{Fa}$ m⁵/S.N | VITESSE MAX. MOTEUR R d/S | C MAINT. ± F C MAX ω=0 $\lambda_1$ | VARIATION DU GAIN $\lambda_2$ | SERVO-VALVES DÉBIT NOM. Q ℓ/mn | COEFF OUVERT.ᵉ $K_s$ mesu m⁶/S√Pa | COEFF FUITES $K_{SOL}$ m⁵/SN | FUITES ASSERV T $\frac{K_{SOL}+K_{Fa}}{2}$ $K_F$ m⁵/SN | RÉDUCTEURS TYPE | RAPPORT N rd/rd (ou rd/m) | INERTIE RAMENÉE À L'AXE $N^2 J_{H.D}$ Kg.m² | CHARGE COTÉ AXE INERTIE DU MASSE J (ou M) Kg.m² (ou Kg) | COUPLE DE MAINTIEN OU FORCE Cm m.N (ou N) | FROTT. SOLIDE SECS F m.N (ou N) | FROTT. VISQUEUX Fv mN/rd/s (ou N/m/s) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | — | SPECIAL | 4,14 | 3,83 | 36,5 | 44 | $2,4.10^{-12}$ | 200 | 0,82 | | 57 | $5,07.10^{-7}$ | $5.10^{-12}$ | $2,5.10^{-12}$ | | 160 | | [illegible] | 0 | 1800 | |
| 2 | | SPECIAL | 4,14 | 3,83 | 36,5 | 44 | $2,4.10^{-12}$ | 250 | 0,46 | | 57 | $5,07.10^{-7}$ | $5.10^{-12}$ | $2.10^{-12}$ | | 160 | | [illegible] | 2200 | 884 | --- |
| 3 | | SPECIAL | 4,44 | 3,93 | 36,5 | 44 | $2,4.10^{-12}$ | 350 | 0,38 | | 67 | $5,05.10^{-7}$ | $5.10^{-12}$ | $2.10^{-12}$ | | 160 | | 845 | 1340 | 2500 | |
| 4 | ROTATION | 4000 T | 4,64 | 4,68 | 40,37 | 46,28 | $4,53.10^{-12}$ | 350 | 0,86 | | 38 | $3,38.10^{-7}$ | $3,38.10^{-12}$ | $4,63.10^{-12}$ | PIGNONS | 43,56 | 0 | 40,6 | 100 | 70 | 30,5 |
| 5 | ROTATION | 4000 T | 4,64 | 4,68 | 40,37 | 46,28 | $4,53.10^{-12}$ | 250 | 0,89 | | 38 | $3,38.10^{-7}$ | $3,38.10^{-12}$ | $169.10^{-12}$ | PIGNONS | 47,586 | 0 | 8,16 | 69 | 74 | 34 |
| 6 | ROTATION | 4000 T | 4,64 | 4,68 | 40,37 | 46,28 | $4,53.10^{-12}$ | 250 | 0,14 | | 38 | $3,38.10^{-7}$ | $3,38.10^{-12}$ | $4,63.10^{-12}$ | PIGNONS | 43,12 | 0 | 2 | 100 | 63 | 30 |

| AXE | ROTATION OU TRANSL. | RAID.ᵀ CÔTÉ AXE HYDRAULIQUE $R_H$ m.N/rd (ou N/m) | RAID.ᵀ CÔTÉ AXE MÉCANIQUE $R_m$ m.N/rd (ou N/m) | PERFORMANCES VITESSE MAX AXE $V_{maxi}$ rd/S (ou m/S) | TEMPS POUR ATTEINDRE VITESSE MAX $T_9$ S | DÉPLACEMENT TYPE déplacement m/m | durée S | bras levier m | VITESSE MAX MOTEUR AU COUPLE MAINT. $\omega_{max}\sqrt{1-\lambda_1}$ rd/S | GAINS ET COEFFICIENTS GAIN ÉLECTRIQUE $K_e$ V/V | GAIN C.N.A $K_{CNA}$ V/ℓ | VALEUR SHUNT MESURE SERVO-VALVE $R_{61}$ Ω | GAIN U/P $K_{UF}$ Hz/V | GAIN C.de MANU $K_C$ V/V | CORRECTEUR R L $f_0$ Hz | CORRECTEUR R L Q max | CODEURS NOMBRE INCRᵀ/TOUR n ℓ/t | C.N.A VOLT/INCRᵀ $K_{CNA}$ V/ℓ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | — | 1235 | | 1 | 1 | 30 | 0,3 | 1 | | 1,1 | $5.10^{-3}$ | 110 | $10^3$ | 1 | 2 | 1,2 | 400 | $5.10^{-3}$ |
| 2 | | 1235 | | 1 | 0,8 | 30 | 0,3 | 1 | | 4 | $5.10^{-3}$ | 110 | $10^3$ | 1 | 2 | 2 | 400 | $5.10^{-3}$ |
| 3 | | 1235 | | 1 | 0,8 | 30 | 0,3 | 1 | | 4 | $5.10^{-3}$ | 110 | $10^3$ | 1 | 8 | 5 | 400 | $5.10^{-3}$ |
| 4 | ROTATION | $4,44.10^6$ | $44.10^6$ | 3 | 0,8 | 30 | 0,3 | 94 - 0,2 | 388,6 | 8 | $5.10^{-3}$ | 400 | $10^3$ | 2,54 | 300 | 200 | 102 | $5.10^{-3}$ |
| 5 | ROTATION | $0,35.10^6$ | $3,5.10^5$ | 3 | 0,8 | 30 | 1,3 | 0,4 | 394,4 | 8 | $5.10^{-3}$ | 400 | $10^3$ | 2,48 | 300 | 200 | 400 | $5.10^{-3}$ |
| 6 | ROTATION | $4,15.10^6$ | $41,5.10^6$ | 3 | 0,8 | 30 | 0,3 | 0,8 | 173,4 | 2 | $5.10^{-3}$ | [illegible] | $10^3$ | 2,53 | 300 | 200 | [illegible] | $5.10^{-3}$ |

Les coefficients du tableau ci contre exprimés en unités MKSA (voir legende du tableau ci dessus) sont obtenus par les formules suivantes dans lesquelles les grandeurs sont considerées :

- n en increments/tour
- Cy en cm³/rd
- Q en ℓ/mn
- Vo en cm³

Unités les plus souvent rencontrées dans les catalogues commerciaux

| | |
|---|---|
| Coeff. fuites S.V. $K_{SOL} = 10^{-5} K_{S\ max}$ | Coefficient d'ouverture $K_{S\ maxi} = 8,9.10^{-9} Q$ |
| Coeff fuites moteur $K_{Fa} = 1,2.10^{-12} V C_y$ | Raideur Hydraulique $R_H = 2500 \frac{C_y^2}{V_0}$ |
| Vit. théorique maxi à couple nul $\omega_{S\ max} = 20 \frac{Q}{C_y}$ | Course d'arrêt d'urgence $\theta_T = \frac{V_{max}^2}{2} \cdot \frac{J}{(1-\lambda) C_{max} N}$ |
| Gain de l'électronique $K_e = \frac{2.10^4}{\sqrt{1-\lambda}} \cdot \frac{C_y}{nQ}$ | Erreur de traînage à V maxi $N_E = \frac{n N V_{max}}{200}$ increment |
| Gain de la comᵈᵉ manuelle $K_c = \frac{\sqrt{1-\lambda}}{10^3 n} \cdot \frac{nQ}{C_y}$ $0 < K_c \leq 1$ | |

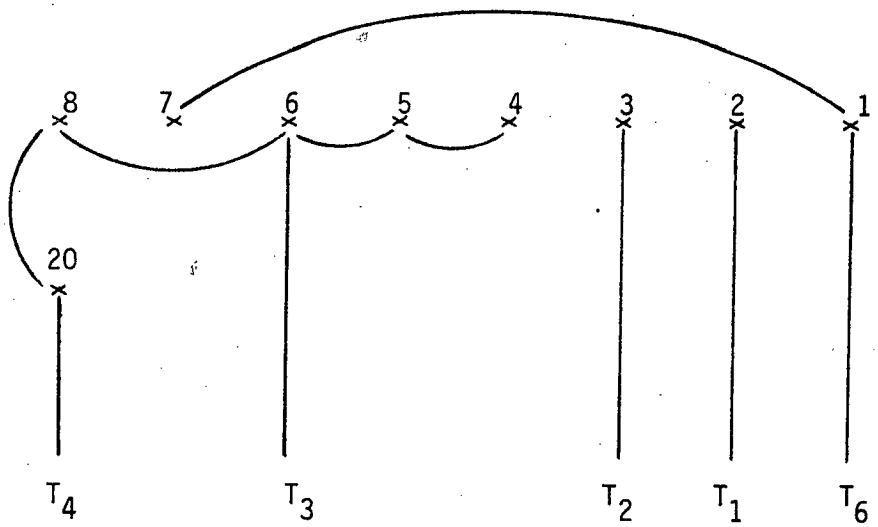
Fig. 12 : Caractéristiques du manipulateur (source RENAULT)

## 1.12. Liaisons série Exormacs - Carte KVM

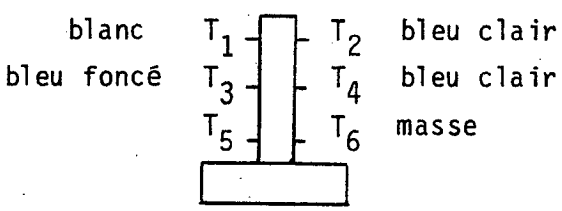
La liaison réalisée est identique à toutes les liaisons téléinformatiques du laboratoire.

Pour mémoire :

Connecteur Subd 25 broches




qui correspond sur la prise type téléphone à :



## 2. Système d'exploitation

C'est lui qui permet un contrôle interactif de l'ensemble robot-carte 68000 < 2 >. Il correspond dans sa version minimum à un moniteur classique étendu de commandes spécifiques au manipulateur et assurant les 3 fonctions suivantes :

- dialogue homme/machine
- commande/contrôle du manipulateur et de la baie
- communication avec un calculateur extérieur.

Il se présente sous la forme d'un noyau temps réel spécifique à 4 niveaux d'exécution qui sont, par ordre de priorité croissante :

- niveau 0 : moniteur ou exécution d'une commande moniteur
- niveau 1 : génération et contrôle des déplacements du manipulateur
- niveau 2 : asservissement des moteurs hydrauliques, sécurités
- niveau 3 : Entrées/Sorties (E/S).

La version 1 du moniteur robot est une version monoprocesseur, ce qui nécessite des E/S prioritaires afin d'assurer la fiabilité de la fonction communication. En contre-partie, des débits d'informations trop importants risquent de perturber les mouvements du manipulateur.

A chacun des niveaux d'exécution correspond une tâche activée par un événement extérieur (interruption, exception, timer ...) ou par une autre tâche (exécution d'une commande moniteur...).

Le synoptique général de fonctionnement du moniteur robot est donné par la fig. 13.

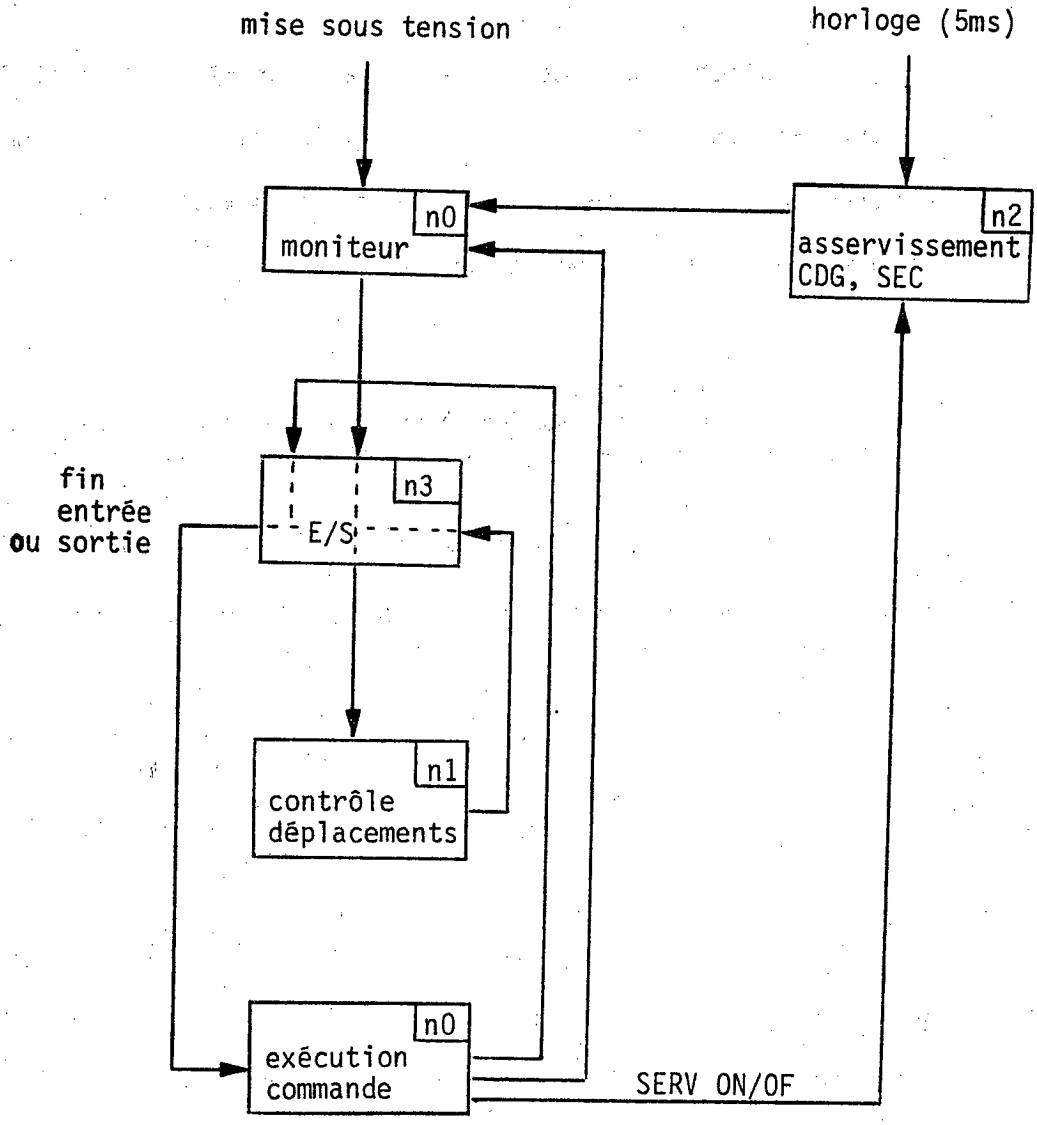



Fig. 13 : Organigramme fonctionnel du système d'exploitation (n = niveau de priorité). Les flèches orientées correspondent à des ordres d'activation de la tâche.

A la mise sous tension, et après avoir affiché un message de bienvenue, le moniteur déclenche une entrée (attente d'une commande de l'opérateur par l'activation de la tâche E/S).

Cette tâche déclenche elle-même périodiquement une autre tâche qui en règle générale assure le contrôle des déplacements du manipulateur (génération de trajectoire, calcul des vitesses et accélérations...). Toute requête au module d'E/S en provenance de la tâche de contrôle des déplacements est prioritaire par rapport à une requête du moniteur.

Une fois l'entrée de la commande moniteur effectuée, celle-ci est exécutée au niveau 0. Elle peut :

- déclencher l'activation de la tâche asservissement
- valider un type de contrôle des déplacements
- déclencher des Entrées/Sorties (affichage, dialogue avec un système hôte ...).

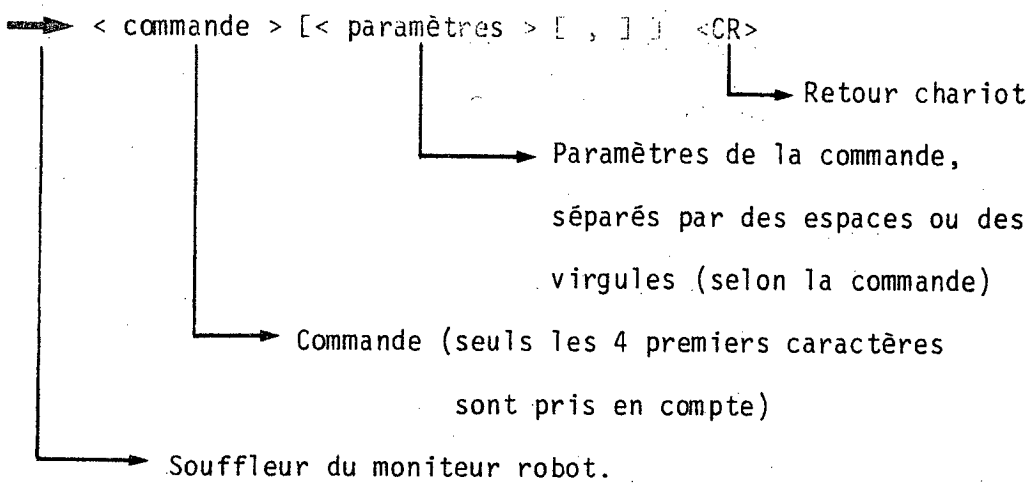
On va maintenant décrire plus en détails les 3 fonctions du moniteur robot.

## 2.1. Fonction dialogue Homme/Machine

Le principe de tout moniteur est d'assurer une communication interactive homme/machine par l'intermédiaire de commandes paramétrées entrées au clavier.

Les commandes existantes sont soit purement informatiques, soit orientées contrôle du manipulateur.

La syntaxe est directement inspirée de la syntaxe Versabug (Motorola) < 3 >.




La seule fonction d'édition actuellement disponible est le bakspace (touche ←).

La touche BREAK provoque un reset du système. Par ailleurs, 4 breaks successifs permettent également un reset lorsqu'un programme est en erreur.

La caractéristique principale de la fonction dialogue est la possibilité d'adaptation de paramètres en temps réel : il est par exemple possible, bien que dangereux, de modifier manuellement les paramètres d'une tâche alors que celle-ci est active.

### 2.1.1. Commandes orientées carte 68000

Elles concernent essentiellement les lectures et écritures de

cases mémoires :

MD affichage mémoire

MS écriture mémoire

MM transfert d'un bloc de mémoire

DC conversion hexadécimal/décimal

HB conversion hexadécimal/binaire.

### 2.1.2. Commandes orientées contrôle du manipulateur

Ce sont les ordres destinés au manipulateur ou à la baie de commande. Ils agissent par l'intermédiaire de la carte couplage robot.

Les commandes actuellement opérationnelles sont :

| | |
|---|---|
| ANGL | affichage des angles mécaniques du manipulateur, en degrés |
| BOUT ON/OF | mise en route/arrêt de la boîte à boutons |
| BOUX ON/OF | idem, mais en mode cartésien |
| BYE | fin de session |
| DEMO | programme de démonstration |
| DOIGTS | description interactive des doigts de la pince |
| ETAT | affichage de l'état du manipulateur (initialisation, positions compteurs...) |
| GO | lancement d'un déplacement en mode libre, point final en positions compteurs |
| GOA | idem, point final, donné en angles mécaniques |
| GOP | lancement d'un déplacement en mode libre, point final (poignet) donné en cartésien |
| GOT | idem, point commandé = point T |
| GOD | lancement d'un déplacement en suivi de ligne droite (interpolation linéaire) |
| UM | contrôle du manipulateur par un calculateur extérieur |
| ORG | rotation du repère de référence (par rapport à un axe vertical) |
| NORG | rétablissement du repère absolu de référence |
| PINC | ouverture/fermeture pince par la boîte à boutons |
| POS | affichage des positions compteurs (en tenant compte de la matrice de couplage) |
| POSP | affichage en mètres et degrés de la position cartésienne du poignet (= XP, YP, ZP, $\psi$, $\theta$, $\phi$) |
| POST | idem, pour le point T |
| RESE | reset |
| SERV ON/OF | mise sous pression puis sous asservissement du manipulateur (ou arrêt) |
| SERV IA/IM | idem, mais avec initialisation automatique (IA) ou manuelle (IM) forcée |
| STOP | arrêt forcé d'un déplacement en cours. |

## 2.2. Fonction commande/contrôle du manipulateur

La structure du noyau temps réel retenue est basée sur la mise en place d'une commande classique de manipulateur à 2 niveaux (fig. 14)

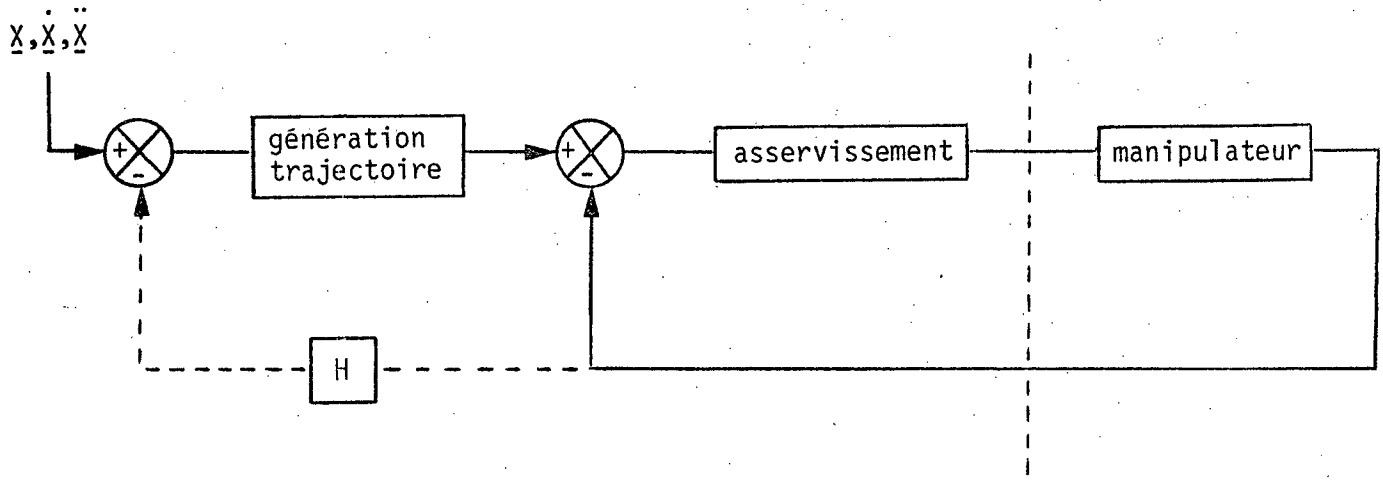



Fig. 14 : Principe de la commande du manipulateur

Le module asservissement permet l'asservissement en position des moteurs hydrauliquess, alors que le module génération de trajectoire assure le calcul des consignes successives nécessaires à l'accomplissement d'un mouvement dont l'objectif et les contraintes ont été spécifiés par un niveau supérieur.

Seule la structure fonctionnelle du noyau temps réel est figée : il est possible, dans une même application, d'affecter aux modules asservissement et contrôle des déplacements des programmes différents permettant des modes de fonctionnement divers (déplacement en mode libre, en ligne droite, asservissement à gains variables ...).

## 2.3. Fonction communication Machine/Machine

Le manipulateur doit s'intégrer dans un environnement, que ce soit pour sa programmation ou son fonctionnement.

Actuellement, les communications sont réalisées par

l'intermédiaire d'une ligne série à 9600 bauds. Pendant la phase de mise au point des programmes elle est reliée au système de développement EXORMACS. Les commandes utiles sont :

TM : mode transparent (utilisation de la console robot comme terminal)

LO : chargement d'un programme à partir de l'EXORMACS (format S).

En fonctionnement, cette ligne est reliée à un mini-ordinateur (Perkin-Elmer 8/32) qui gère les capteurs de vision par l'intermédiaire du langage de programmation LM < 4 >. Par la commande

LM = passage sous le contrôle de LM

le moniteur robot vient prendre ses ordres sur la ligne série 2. Disposant d'un ensemble de primitives implantées sur l'unité centrale du robot, le calculateur extérieur peut alors commander ou interroger le manipulateur.

## 2.4. Génération d'un système d'exploitation

Compte-tenu de la souplesse d'évolution requise par toute activité de recherche, tout le logiciel a été développé de façon modulaire. Les objectifs poursuivis par l'organisation mise en place sont :

- construction d'applications spécifiques, permettant d'optimiser l'occupation mémoire et le temps de développement
- standardisation des modules, avec rangement dans des bibliothèques communes à tous les utilisateurs
- mise en place de fichiers de commande assurant l'assemblage et l'édition de liens d'une application.

En règle générale, une application spécifique est obtenue en configurant le fichier TABCOM.SA avec les seules commandes moniteur nécessaires. Les nouveaux programmes ou les versions corrigées sont incluses au moment de l'édition de liens.

Le fichier de commande LNKV.CF assure l'édition de liens avec scrutation des bibliothèques et l'organisation de l'image mémoire. Le fichier de sortie est un fichier en format S directement chargeable sur la carte KVM (voir chapitre 4.).

## 2.5. Utilitaires du système d'exploitation

Ce sont tous les programmes nécessaires ou utiles à l'implantation du moniteur robot et de ses programmes d'application.

### 2.5.1. Drivers d'Entrées/Sorties < 2 >

Des drivers d'E/S fonctionnant en interruptions ont du être développés pour chacun des 2 ports série de la carte unité centrale.

Le 1$^{er}$ est destiné à gérer la console opérateur et le 2$^{ème}$ une liaison avec un calculateur : les modes de fonctionnement ne sont donc pas identiques. Ils sont décrits dans < 2 >.

a - Driver console opérateur

Il doit permettre à l'opérateur de reprendre le contrôle au clavier à tout moment.

Toute touche frappée au clavier génère une interruption de niveau 5. Plusieurs modes de fonctionnement existent :

Mode 0 : ni lecture, ni écriture
Mode 1 : lecture
Mode 2 : sortie
Mode 6 : sortie suspendue
Mode 7 : mode transparent.

L'interruption est déclenchée indifféremment par une entrée (registre de donnée plein) ou par une sortie (port libre).

Un compteur de breaks déclenche un reset du système si la touche break a été activée 4 fois.

Le CTRL-W permet de suspendre une sortie, qui est relancée par n'importe quelle autre touche.

b - Driver liaison calculateur

L'interruption port 2 est également branchée sur le niveau 5. Les échanges sont effectués essentiellement sur compte d'octets ou caractère de fin d'échange, à partir d'un buffer circulaire de 150 octets. Un flag de validation, positionné par TRAP # 5, permet d'exécuter ou de refuser des transferts d'octets sur le port 2.

2.5.2. Handlers d'E/S

Ils servent d'interface entre le programme utilisateur et le driver d'E/S. Ils sont chargés de déclencher, d'entretenir ou d'annuler des E/S. Ils sont activés à partir d'un programme utilisateur par l'intermédiaire de l'instruction TRAP du 68000.

a - Trap # 7 : E/S console opérateur

A chaque appel, l'adresse d'un bloc de contrôle est transmise au handler dans le registre A0.

Le bloc de contrôle est donné par la fig. 15.

| | | |
|---|---|---|
| BLOC | DC.B FONCTION | 0 : entrée avec écho<br>1 : sortie<br>2 : arrêt de l'échange en cours<br>$10 : mode transparent |
| STATUS | DS.B 1 | $FF : échange en cours<br>$00 : échange terminé sur RETURN<br>$01 : échange terminé sur DEL<br>$02 : échange terminé (fin de buffer)<br>$0F : échange terminé sur CANCEL<br>$80 : erreur dans le paramètre FONCT.<br>$81 : erreur dans les adresses<br>$8F : échange déjà en cours |
| | DC.L BUFFER | adresse du buffer |
| | DC.L FINBUF | adresse de la fin du buffer |
| GR | DS.W 1 | longueur échangée |

Fig. 15 : Bloc de contrôle TRAP # 7 (E/S console)

b - TRAP # 8 : E/S calculateur extérieur

Le principe est le même que pour le TRAP # 7. Le bloc de contrôle est donné par la fig. 16.

| | | |
|---|---|---|
| BLOC | DC.B FONCTION | 0 : lecture buffer<br>1 : écriture buffer<br>$80 : test buffer d'entrée<br>$81 : test buffer de sortie |
| | *** si FONCTION = 0 ou 1 *** | |
| STATUS | DS.B 1 | $FF : échange en cours (sortie)<br>$00 : échange terminé<br>$01 : port 2 inhibé<br>$81 : port 2 validé<br>$80 : fonction illégale<br>$8F : échange déjà en cours (sortie)<br>$88 : buffer pas assez plein (entrée) |
| | DC.L BUFFER | adresse buffer |
| | DC.W LONGUEUR | longueur à échanger |
| | *** si FONCTION = $80 *** | |
| STATUS | DS.B 1 | contient 0 ou le premier caractère du buffer d'entrée ou $FF si RETURN reçu |
| LGREC | DS.W 1 | contient la longueur reçue |
| | *** si FONCTION = $81 *** | |
| STATUS | DS.B 1 | $8F = échange en cours<br>0 = échange terminé |
| LGREC | DS.W 1 | contient la longueur reçue |

Fig. 16 : Bloc de contrôle TRAP # 8 (E/S port 2)

2.5.3. TRAP # 5 : utilitaires

Il permet l'exécution de fonctions utilitaires liées aux E/S. Le numéro de la fonction est passé dans D0.B. La réponse, si elle existe, se trouve également dans D0. Les fonctions disponibles sont :

D0 = 0 : Le compteur de breaks est transféré dans D0, puis remis à zéro.

D0 = 1 : A0 contenant l'adresse d'un bloc de paramètres, une E/S est lancée sur la console opérateur (TRAP # 7) et la fin de l'échange

attendue (status à $FF). Cette requête active également périodiquement la tâche contrôle des déplacements du moniteur robot.

D0 = 2 : C'est la même requête que 1, l'échange pouvant être interrompu par un break.

D0 = 3 : Requête identique à 1, mais sans activation de la tâche contrôle des déplacements.

D0 = 4 : Attente de fin d'échange sur le port 2 avec activation de la tâche contrôle des déplacements et retour sur break.

D0 = 5 : Attente de la fin d'exécution de la tâche contrôle des déplacements.

D0 = 6 : Attente du caractère de fin de mode transparent (CTRL-A).

D0 = $10 : A0 contient l'adresse de la tâche à exécuter lors de l'activation du module de contrôle des déplacements.

D0 = $20 : Activation de la tâche définie avec le code $10.

D0 = $80 : Invalidation du port 2.

D0 = $81 : Validation du port 2.

## 2.6. Initialisation - Implantation mémoire

Le programme d'initialisation du driver doit être lancé en $1800.

Ce programme se branche ensuite en $1C00 qui est le début du programme moniteur robot.

L'implantation mémoire retenue est :

0 - $3FF : table des vecteurs d'exceptions
$400 - $7FFF : pile du moniteur robot
$1000 - $17FF : driver + handlers d'E/S série
$1800 - $19FF : initialisation système
$1A00 - $1BFF : variables driver + système
$1C00 - $FFFF : programmes + variables d'application

2.7. Utilitaires de conversion (tableau 17)

Ce sont tous les sous-programmes assurant des conversions. Ils sont regroupés sous le terme de traducteurs, dont les numéros 1 à 4 sont d'intérêt général que les autres sont plus spécifiques à l'application robotique.

2.8. Sous-programme divers (tableau 18)

Ils rendent accessibles les diverses fonctions système (appels aux handlers d'entrées/sorties ...) à des programmes d'application par l'intermédiaire de procédures simples.

Ce sont essentiellement des sous-programmes :

- d'entrées/sorties
- de temporisation
- de décodage d'une ligne de commande.

| NOM | Fonction | Entrée | Sortie | Compte-Rendu |
|---|---|---|---|---|
| TRADU1 | Conversion Entier/Ascii | D0.L = entier long à convertir<br>A0.L = @ de rangement<br>A1.L = @ fin de buffer de rangement | A2.L = @ du 1er octet n'appartenant pas au nombre | |
| TRADU2 | Conversion Ascii/Entier | A0.L = @ début de chaîne | D0.L = entier converti | D1.B = 0 si OK |
| TRADU3 | Conversion Hexa/Ascii | D0.L = nombre hexa à convertir<br>D1.B = nombre de digits en sortie<br>A0.L = @ début de chaîne | A2.L = @ du 1er octet n'appartenant pas au nombre | |
| TRADU4 | Conversion Ascii/Hexa | A0.L = @ début chaîne à convertir | D0.L = nombre converti<br>A0.L = @ dernier caractère de la chaîne | D1.B = 0 si OK |
| TRADU5 | Conversion positions compteurs/ angles mécaniques | A0.L = @ tableau des positions compteurs<br>A1.L = @ tableau des angles mécaniques | tableau des angles mécaniques rempli | |
| TRADU6 | Conversion longueur 32 bits (codage spécifique)/chaîne Ascii | A0.L = @ nombre 32 bits à convertir<br>A1.L = @ début de chaîne | chaîne Ascii | |
| TRADU7 | Conversion longueur Ascii décimale/32 bits (codage spécifique) | A0.L = @ début de chaîne à convertir | D0.L = longueur codée sur 32 bits | D1.B = 0 si OK<br>= 1 si paramètre par défaut<br>= 2 si longueur > 7 mètres<br>= 3 si caractère non numérique<br>= 4 si plus de 4 décimales<br>= 5 si erreur de formatage du nombre Ascii |
| TRADU8 | Conversion angles mécaniques/ positions compteurs | A0.L = @ tableau angles mécaniques<br>A1.L = @ tableau positions compteurs | . tableau des consignes moteurs rempli<br>. limitation systématique en cas de dépassement | D1.L = 0 si OK<br>bit 2 * (i-1) = 1 ⟹ limite max sur axe i<br>bit (2 * (i-1)) + 1 = 1 ⟹ limite min sur axe i |
| TRADU9 | Conversion angles frationnaire Ascii/16 bits (codage spécifique) | A0.L = @ début de chaîne à convertir | D0.W = angle codé sur 16 bits | D1.B = 0 si OK<br>= 1 si paramètre par défaut<br>= 2 si ABS (angle) > 180°<br>= 3 si caractère non numérique<br>= 4 si plus de 2 décimales<br>= 5 si erreur de formatage |
| TRADU10 | Conversion angle 16 bits (codage spécifique)/chaîne Ascii | A0.L = @ nombre 16 bits à convertir<br>A1.L = @ début de chaîne | | |
| TRADU11 | Conversion d'un SIN ou COS Ascii/ 16 bits (codage spécifique) | A0.L = @ de la chaîne à convertir | D0.W = SIN/COS codé sur 16 bits | D1.B = 0 si OK<br>= 1 si paramètre par défaut<br>= 2 si erreur |
| TRADU12 | Conversion d'un SIN ou COS sur 16 bits (codage spécifique)/ chaîne Ascii | A0.L = @ du SIN ou COS à convertir<br>A1.L = @ de la chaîne | | |

Tableau 17 : Utilitaires de conversion

| NOM | Fonction | Entrée | Sortie | Compte-Rendu |
|---|---|---|---|---|
| LECMESS | Lecture d'une chaîne de caractères au clavier<br>Fin d'échange sur :<br>. fin de buffer<br>. return<br>. touche del | A0.L = @ du buffer à remplir<br>A1.L = @ de fin du buffer à remplir | . buffer conditionné<br>. D2.W = longueur échangée<br>. attente de fin d'échange avant le retour | D0.B = 0 OK<br>= 1 si break<br>= 2 si erreur |
| LECCAR | Point d'entrée de LECMESS pour la lecture d'un caractère au clavier | | D1.B = caractère Ascii lu | |
| ECRIMESS | Ecriture d'une chaîne de caractères Ascii terminée par $ 14 sur l'écran | A0.L = @ de la chaîne Ascii à afficher | . A0.L et D0.W modifiés<br>. retour après initialisation de l'échange | |
| ECRMESS2 | Ecriture d'une chaîne de caractères Ascii terminée par $ 0 (return) sur le port 2 | | . A0.L et D0.W modifiés<br>. retour après initialisation de l'échange | |
| ECRMESST | Identique à ECRIMESS, sauf que la requête provient d'une tâche de contrôle des déplacements | A0.L = @ de la chaîne Ascii à afficher | . A0.L et D0.W modifiés<br>. retour après initialisation de l'échange | |
| TEMPO | Temporisation programmable | D0.W = durée de la tempo.<br>(D0 = 50 ms) | | Attention : ce S.P. utilisant le timer, ce dernier doit avoir été initialisé par INIHTR |
| UMONIT | Contient les utilitaires des commandes moniteur Minibug empruntés à MOTOROLA (affichage ou modification cases mémoires....) | | | |

Tableau 18 : Sous-programmes d'intérêt général

## 3. Commande/Contrôle du manipulateur

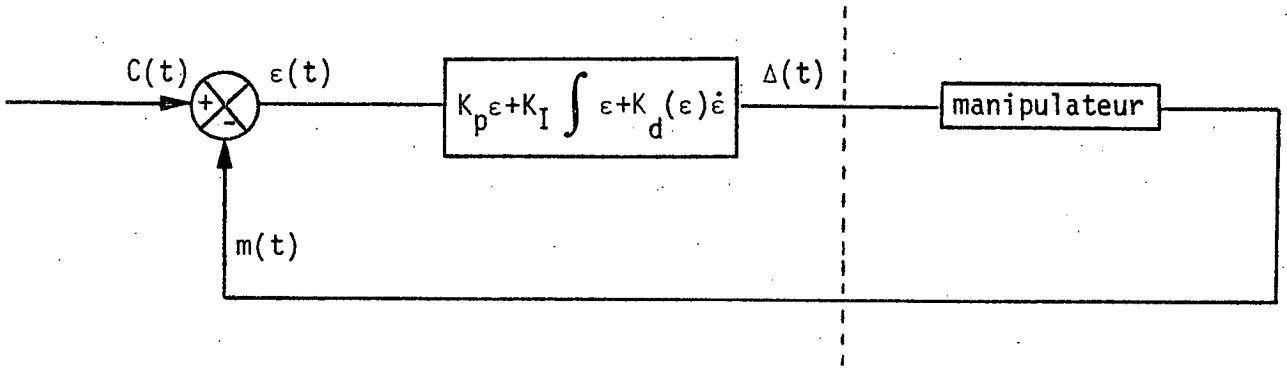
Ce chapitre présente les différents modules nécessaires à la réalisation de la commande du manipulateur.

### 3.1. Asservissement

Le premier asservissement réalisé est un asservissement de type proportionnel-intégral-dérivée (PID). La fréquence d'échantillonnage retenue est 200 Hz (5 ms), ce qui, comparé aux quelques Hertz (8 expérimentalement) de bande passante du manipulateur, permet de négliger les effets de la discrétisation.

Le terme intégral du correcteur permet d'annuler l'erreur de position finale, due essentiellement aux frottements secs et au décalage du zéro des servo-valves.

Le schéma de principe de l'asservissement de chacun des moteurs hydrauliques est donné par la fig. 19.




C(t) = consignes en position

m(t) = positions réelles

ε(t) = écart de position

s(t) = consignes moteurs

Fig. 19 : Schéma de principe de l'asservissement en position des moteurs hydrauliques

En continu, on a :

$$s(t) = K_p\ \varepsilon(t) + \frac{1}{K_I}\int_0^t \varepsilon(t)\ dt + K_d(\varepsilon)\ \dot{\varepsilon}(t)$$

En discrétisant, et en intégrant par la méthode des trapèzes, on obtient :

$$s_k = K_p\ \varepsilon_k + \frac{1}{K_I} \sum_{0}^{k} \frac{\varepsilon_j + \varepsilon_{j-1}}{2} + K_d\,(\varepsilon_k - \varepsilon_{k-1})$$

Pour obtenir une formule récursive, on pose :

$$s_k - s_{k-1} = (K'_p(\varepsilon_k - \varepsilon_{k-1}) + \frac{1}{2K_I}(\varepsilon_k + \varepsilon_{k-1})) + K'_d(\varepsilon_k - 2\varepsilon_{k-1} + \varepsilon_{k-2})$$

$$\boxed{s_k = K_p(\varepsilon_k - \varepsilon_{k-1}) + K'_I(\varepsilon_k + \varepsilon_{k+1}) + K'_d(\varepsilon_k - 2\varepsilon_{k-1} + \varepsilon_{k-2})}$$

avec $K'_I = \frac{1}{2K_I}$ et $K'_d = \frac{K_d \times \varepsilon}{Vmax}$ (proportionnel à la vitesse).

Pour ne pas déstabiliser en cours de mouvement, le terme intégral peut-être inhibé par l'intermédiaire d'un flag. Par ailleurs, même s'il est validé, il ne se déclenche que si l'erreur de position finale est faible.

Les variables programmées sont :

$s_k$ = COMD

$\varepsilon_k$ = ECPOS

$\varepsilon_{k-1}$ = ECPOSP  d'où :

- COMD = $K_P$(ECPOS-ECPOSP) + $K'_I$(ECPOS+ECPOSP) + $K'_D$(ECPOS-2*ECPOSP+ECPOSPP) + COMD
- $K'_D = K_D$ x 4 x ECPOS / VMAX

Cet algorithme très simple ne nécessite que 6x(4 additions + 2 soustractions + 6 multiplications + 1 division) pour une erreur de positionnement finale inférieure ou égale à 1 incrément compteur.

Les gains proportionnels de chacun des moteurs ont été ajustés légèrement en dessous de l'instabilité, alors que les gains du terme

intégral assurent une erreur finale presque nulle, mais sans dépassement de consigne.

Les points restant à étudier sont :

. la correction des erreurs de traînée, peut-être par l'utilisation d'une commande auto-adaptative < 5 >

. la définition de gains variables, selon la configuration ou le mouvement du manipulateur

. .....

## 3.2. Contrôle des déplacements

C'est le module chargé de générer les consignes des asservissements. Son rôle est avant tout de générer et contrôler la trajectoire du manipulateur. Etant placé en amont des asservissements (fig. 20), et compte-tenu de la masse de calculs à effectuer, la fréquence d'activation de cette tâche est avant tout fixée par des contraintes technologiques : selon le type de trajectoire, elle peut dans notre cas, se situer entre 50 Hz (20 ms) et 100 Hz (10 ms)

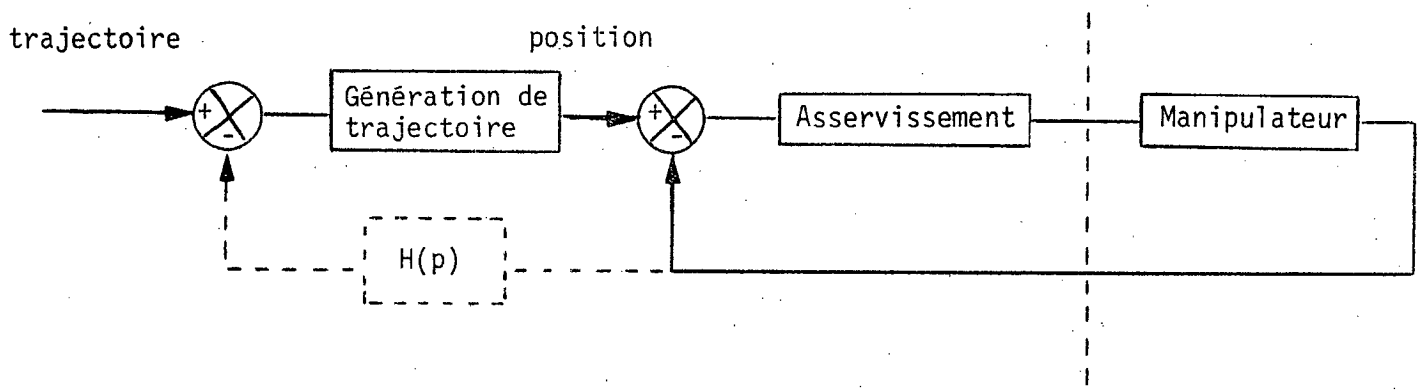



Fig. 20 : Différents niveaux de la commande du manipulateur

2 types de générateurs de trajectoire sont actuellement opérationnels :

. le déplacement en mode libre

. le déplacement en ligne droite.

3.2.1. Déplacement en mode libre < 6 >

Développé au CERT, tous les détails de l'algorithme sont donnés dans < 6 >.

Dans un déplacement de ce type, les seules contraintes sont la

position finale à atteindre et la vitesse maxi autorisée. Les déplacements sur chacun des axes sont synchronisés de sorte à assurer une arrivée à vitesse nulle simultanée sur tous les axes. La trajectoire est divisée en 3 parties :

- déplacement à accélération constante
- déplacement à vitesse constante
- freinage à accélération constante.

La trajectoire étant planifiée par avance, le cumul des erreurs de calcul entraîne, notamment lors de grands déplacements, une erreur de position finale importante qu'il est nécessaire de rattraper avant de déclencher l'action intégrale.

Bien que rudimentaire, le déplacement en mode libre est le plus utilisé : en effet, les contraintes de trajectoire ne sont nécessaires que dans des cas particuliers (évitement d'obstacle, suivi de joint de soudure, ...).

3.2.2. Déplacement en ligne droite < 7 > < 8 >

Dans ce cas, le déplacement est spécifié dans le repère cartésien puis traduit en déplacements dans le repère des coordonnées généralisées du manipulateur par l'intermédiaire du modèle géométrique.

Une correction est introduite en rebouclant sur la commande la position cartésienne réelle, permettant ainsi une régulation sur la droite de consigne.

De même que pour le déplacement en mode libre, la trajectoire est

découpée en 3 parties dans le temps :

$0 < t < T1 \qquad \gamma = \gamma_{max}$

$T1 < t < T2 \qquad \gamma = 0$

$T2 < t < T \qquad \gamma = -\gamma_{max}$

Avec une contrainte de temps complet de déplacement, on obtient :

$$T1 = \frac{T}{2} - \sqrt{\frac{T^2}{2} + \frac{4U}{\gamma_{max}}}$$

$$T2 = T - T1$$

d'où on déduit $v_{max-mvt}$ = vitesse maxi atteinte au cours du mouvement, qui doit être impérativement inférieure à $v_{max}$.

Tous les détails de cet algorithme et de son implémentation sont donnés dans < 7 > et < 8 >.

Une méthode pour la validation de la trajectoire sans la calculer y est également décrite : elle permet d'éviter de lancer un déplacement impossible pour le manipulateur, notamment si la droite demandée traverse des butées mécaniques.

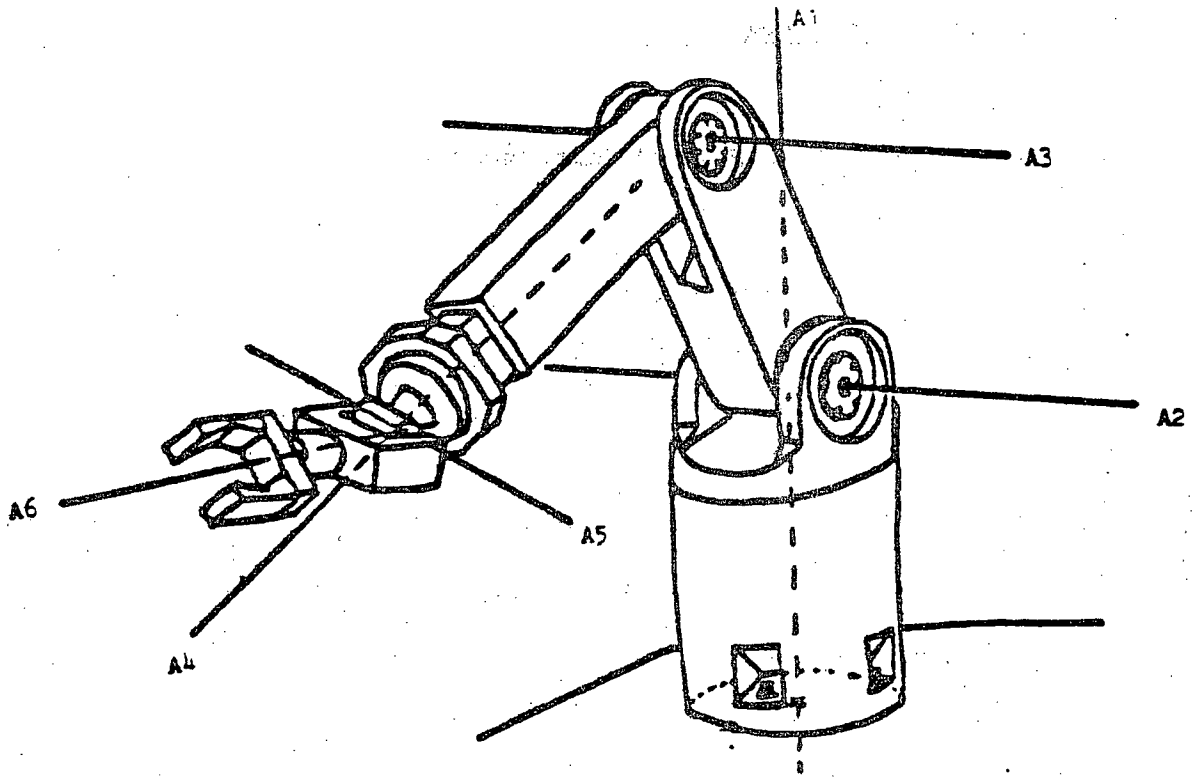
Une implantation en PASCAL de ces algorithmes est prévue : l'emploi d'un opérateur (câblé ou programmé) flottant devrait permettre de rendre négligeables les erreurs dues au calcul préliminaire des incréments de consigne unitaire.

Les performances relevées sont : temps d'exécution de la tâche = 8,4 ms (2 transformations de coordonnées), d'où une possibilité d'échantillonnner à 10 ms.

## 3.3. Modèle géométrique

### 3.3.1. Formulation

On appelle $q_1$, $q_2$, $q_3$, $q_4$, $q_5$, $q_6$ les angles des articulations du robot ou coordonnées généralisées et X, Y, Z, $\psi$, $\theta$, $\phi$ l'attitude de l'organe terminal (3 coordonnées de position, 3 coordonnées d'orientation). Le modèle géométrique est l'ensemble des équations qui permettent de passer des coordonnées de configuration aux coordonnées opérationnelles et inversement. < 9 > (fig. 21)




Schéma

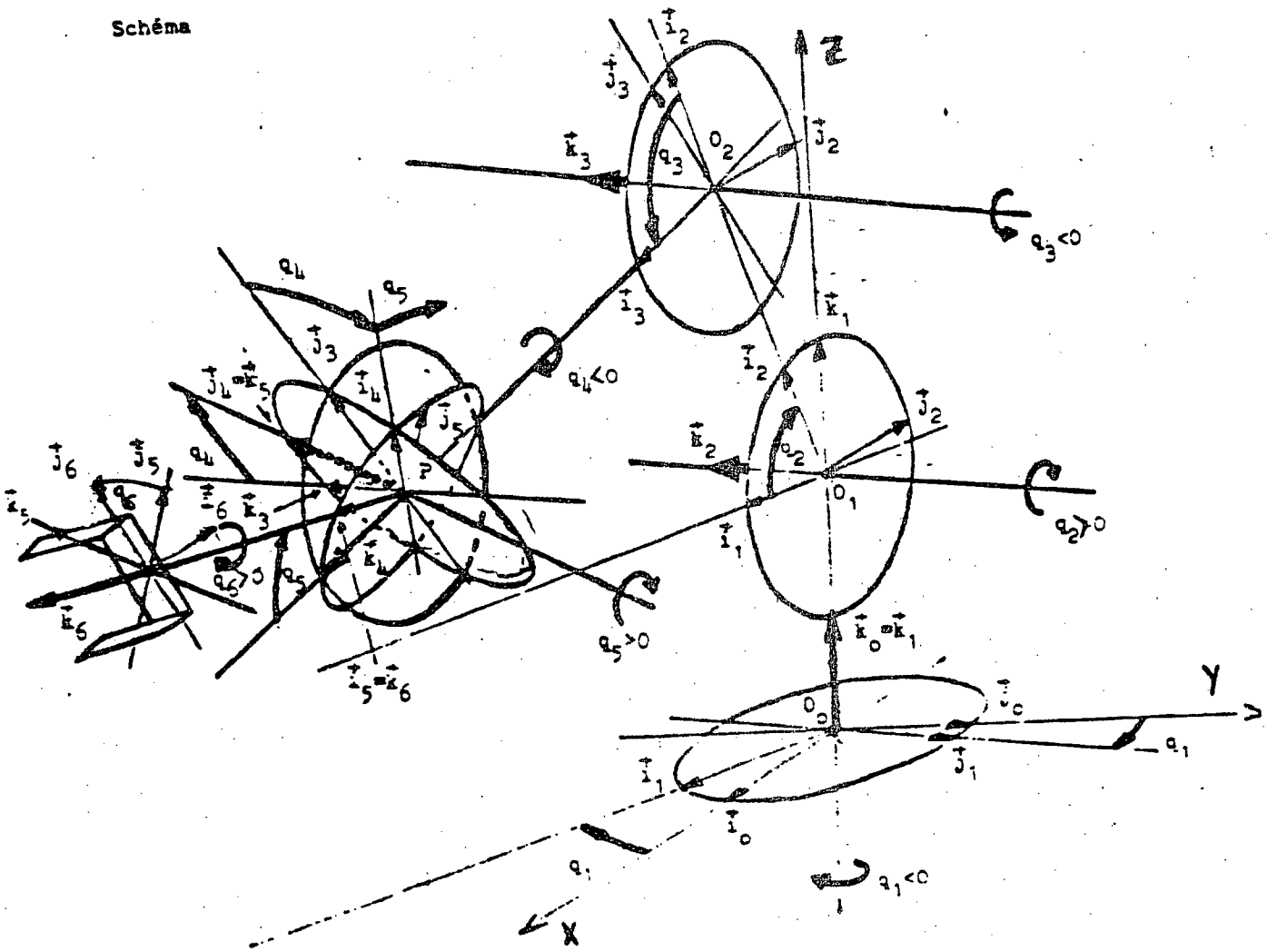



Fig. 21 : Description du manipulateur V80 < 9 >

Modèle direct

Il est donné par les équations :

$$(1) \quad \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} C_1(l_2C_2 + l_3C_{23}) \\ S_1(l_2C_2 + l_3C_{23}) \\ l_2S_2 + l_3S_{23} \end{pmatrix}$$

et

$$(2) \quad \begin{pmatrix} \psi \\ \theta \\ \phi \end{pmatrix} = \begin{pmatrix} \text{arctg } a_{23}/a_{13} \\ \text{arcsin } a_{33} \\ \text{arctg } -a_{31}/-a_{32} \end{pmatrix}$$

où

$$a_{33} = S_{23}C_5 + C_{23}C_4S_5$$
$$a_{23} = S_1(C_{23}C_5 - S_{23}C_4S_5) - C_1S_4S_5$$
$$a_{13} = C_1(C_{23}C_5 - S_{23}C_4S_5) + S_1S_4S_5$$
$$a_{32} = -S_{23}S_5C_6 + C_{23}(C_4C_5C_6 - S_4S_6)$$
$$a_{31} = -S_{23}S_5S_6 + C_{23}(C_4C_5C_6 + S_4C_6)$$

Modèle inverse

L'inversion se fait en 2 temps. D'abord on inverse le modèle du poignet, puis avec les valeurs retenues pour $q_1$ $q_2$ $q_3$, on inverse le modèle de l'orientation pince.

On obtient :

$$(3)\quad \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} \text{arctg } Y/X \\ \alpha + \beta \\ -2_\beta \end{pmatrix} \text{ avec } \quad \alpha = \text{arctg } Z/\sqrt{X^2 + Y^2}$$

$$\beta = \arccos\left(\frac{\sqrt{X^2+Y^2+Z^2}}{2}\right)$$

puis :

$$(4)\quad \begin{pmatrix} q_5 \\ q_4 \\ q_6 \end{pmatrix} = \begin{pmatrix} \arccos b_{13} \\ \text{arctg } b_{33}/b_{23} \\ \text{arctg } -b_{11}/-b_{12} \end{pmatrix}$$

où $b_{ij}$ sont les éléments de la matrice produit $S_{03}^t SA$, avec :

$$S_{03}^t = \begin{pmatrix} C_1C_{23} & S_1C_{23} & S_{23} \\ -C_1S_{23} & -S_1S_{23} & C_{23} \\ S_1 & -C_1 & 0 \end{pmatrix}$$

et

$$S_A = \begin{pmatrix} -C_\psi S_\theta S_\phi - S_\psi C_\phi & -C_\psi S_\theta C_\phi + S_\psi S_\phi & C_\psi C_\theta \\ -S_\psi S_\theta S_\phi + C_\psi C_\phi & -S_\psi S_\theta C_\phi - C_\psi S_\phi & S_\psi C_\theta \\ C_\theta S_\phi & C_\theta C_\phi & S_\theta \end{pmatrix}$$

3.3.2. Difficultés de mise en oeuvre des modèles géométriques

a - Singularités

Le modèle direct présente une singularité en $\theta = \pm \frac{\Pi}{2}$ où $\psi$ et $\phi$ ne sont plus séparables.

De même, le modèle inverse présente une indétermination sur $q_4$ et $q_6$ lorsque $C_5 = 1$, c'est-à-dire lorsque $q_5 = 0$. Dans ce cas, il est possible de calculer $q_4 + q_6$. Néanmoins, 3 problèmes importants apparaîssent lors de l'implantation sur microprocesseur :

- les effets dus à la singularité (précision des calculs) se font sentir dans un intervalle autour de cette singularité

- le calcul de $q_4$ et $q_6$ nécessite de fixer arbitrairement une de ces 2 variables (au détriment d'une perte d'orientation)

- le choix de $q_4$ et $q_6$ est directement lié au choix d'une des 8 solutions possibles pour les coordonnées généralisées.

b - Temps de calcul

Le volume des calculs à effectuer doit tenir compte des impératifs temps réel de la commande du robot. On estime que pour le suivi de ligne droite, la consigne doit être recalculée tous les 20 ms environ. Un véritable asservissement en ligne droite nécessitant une transformation de coordonnées inverse dans la boucle directe et une transformation directe dans la boucle de retour, l'objectif à atteindre est un temps de calcul inférieur à 20 ms pour les 2 modèles.

c - Précision

Cette rapidité nécessaire doit se faire dans des conditions de précision acceptables. Actuellement, les meilleurs robots industriels 6 axes (soudure, manutention) sont garantis pour une répétabilité de l'ordre de 2/10 mm. La précision du positionnement de l'organe terminal dépend directement de la précision du modèle géométrique et de son calibrage par rapport à une référence absolue.

Le robot n'étant pas destiné, comme les machines outil, à usiner des pièces, on considèrera dans un premier temps qu'une précision de positionnement de l'ordre de 1/10 mm est suffisante.

Calculs d'erreurs - Estimation des incertitudes

En assimilant l'erreur absolue à la différentielle totale, les équations (1) donnent :

$$dX = \frac{\partial X}{\partial C_1} dC_1 + \frac{\partial X}{\partial C_2} dC_2 + \frac{\partial X}{\partial C_{23}} dC_{23} \quad \text{avec } l_2 = l_3 = 1$$

$$dX = (C_2+C_{23})dC_1 + C_1 dC_2 + C_1 dC_{23}$$

Or :

$$C_1 = \cos q_1 \Rightarrow dC_1 = \frac{\partial C_1}{\partial q_1} dq_1 = -\sin q_1 dq_1 = -S_1 dq_1$$

de même :

$$C_{23} = -S_{23}(dq_2+dq_3)$$

On en déduit :

(5)

$$\boxed{\begin{aligned} dX &= -S_1(C_2+C_{23})dq_1 - C_1S_2dq_2 - C_1S_{23}(dq_2+dq_3) \\ dY &= C_1(C_2+C_{23})dq_1 - S_1S_2dq_2 - S_1S_{23}(q_2+dq_3) \\ dZ &= C_2dq_2 + C_{23}(dq_2+dq_3) \end{aligned}}$$

avec dX, dY et dZ = erreurs absolues sur X, Y et Z.

On définit l'incertude absolue comme un majorant de l'erreur absolue :

$$\Delta u \geq |du|$$

En prenant les cas les plus défavorables ($S_i = C_i = 1$)

$$\Delta X = 2\Delta q_1 + 2\Delta q_2 + \Delta q_3$$

$$\Delta Y = 2\Delta q_1 + 2\Delta q_2 + \Delta q_3$$

$$\Delta Z = 2\Delta q_2 + \Delta q_3$$

Ce qui entraîne comme conditions sur $\Delta q_1$, $\Delta q_2$, et $\Delta q_3$ :

$$2\Delta q_1 + 2\Delta q_2 + \Delta q_3 \leq 10^{-4}$$
$$2\Delta q_1 + \Delta q_3 \leq 10^{-4}$$

donc :

(6)
$$\boxed{\begin{aligned} \Delta q_1 &\leq \frac{10^{-4}}{4} \text{ rds} \\ \Delta q_2 &\leq \frac{10^{-4}}{3} \text{ rds} \\ \Delta q_3 &\leq \frac{10^{-4}}{3} \text{ rds} \end{aligned}}$$

Ces contraintes imposées pour le calcul des 3 premiers angles fournissent une indication sur la précision requise par les fonctions mathématiques (équation (3)).

En outre, et à condition que les calculs intermédiaires soient effectués à un ordre supérieur, la précision de l'orientation sera du même ordre de grandeur que les fonctions trigonométriques (équations (4)), c'est-à-dire $10^{-4}$ rds = $5.7\ 10^{-3}$ degré.

Les différentes techniques retenues pour l'implémentation en langage Assembleur vont maintenant être décrites.

### 3.3.3. Codage des variables

Le codage des variables, c'est-à-dire la représentation interne des données est un compromis entre le type de processeur (16 ou 32 bits), la précision requise pour les calculs et le type d'instructions mathématiques disponibles ainsi que leurs temps d'exécution.

Bien qu'il dispose de registres internes 32 bits, le 68000 doit être considéré comme un 16 bits. En effet, les échanges avec les mémoires se font sur un bus de données 16 bits.

Les types de données à coder sont :

- les longueurs
- les angles
- les valeurs des fonctions trigonométriques.

Le codage le plus évident d'une grandeur x sur 16 bits serait :

$X_{min}$ = \$8000

si $X_{min} < 0$ et $X_{max} > 0$

$X_{max}$ = \$7FFF

sinon

$X_{min}$ = \$0

$X_{max}$ = \$FFFF

a - Codage des angles

Les butées mécaniques du Robot V80 permettent approximativement les débattements suivants (les limites réelles étant données par la fig. 22) :

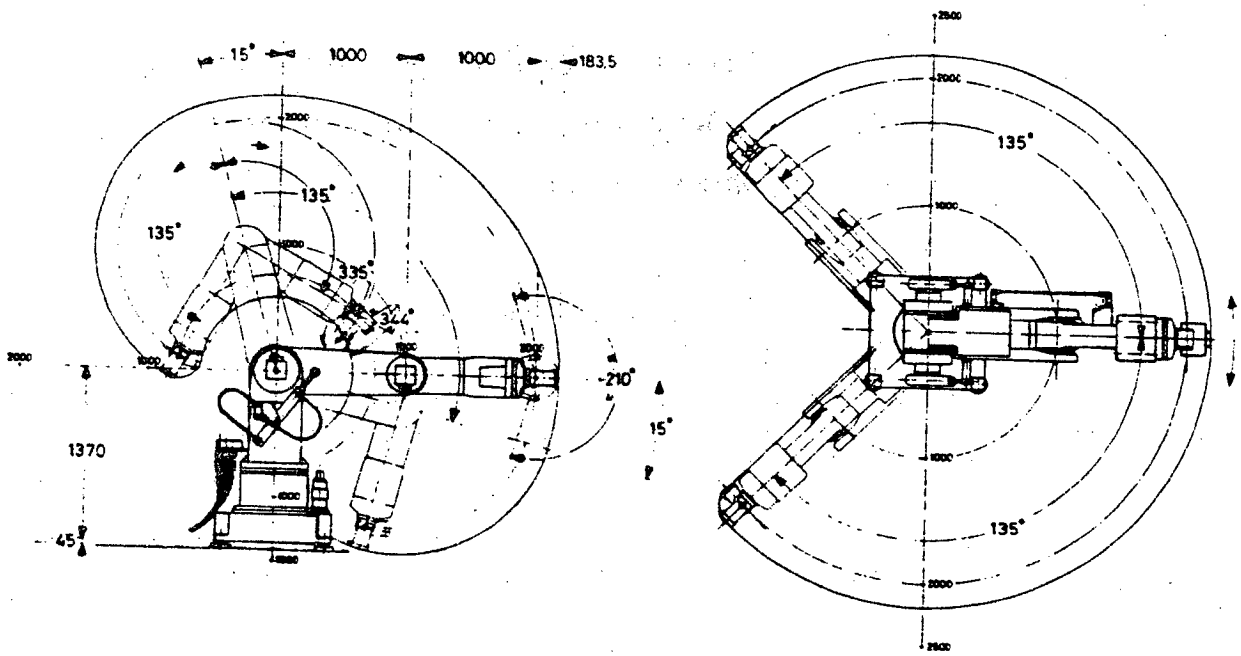



Fig. 22 : Limites angulaires du manipulateur V80

(notice constructeur)

| angle méca | min(d°) | max(d°) |
|---|---|---|
| $q_1$ | -105 | +105 |
| $q_2$ | -15 | +105 |
| $q_3$ | -133 | +115 |
| $q_4$ | -152 | +152 |
| $q_5$ | -84 | +77 |
| $q_6$ | -152 | +152 |

Ces valeurs ne sont pas figées ; en particulier $q_4$ et $q_6$ n'ont pas de valeurs limites (rotations infinies).

Afin d'éviter un codage spécifique à chaque angle, on choisit classiquement :

-180° = $8000

+180° = $7FFF

En réalité, la valeur exacte +180° n'a pas de représentation interne.

L'intérêt d'un tel codage est que la périodicité (angles définis à 2 Π près) résoud automatiquement les problèmes d'overflow pouvant survenir au cours de calculs intermédiaires.

Il faut néanmoins émettre certaines réserves sur cet intérêt : si, sur un plan purement mathématique, -180° = +180° il n'en est pas de même pour un manipulateur...

La résolution minimale que l'on obtient est :

$$\frac{\Pi}{32768} = 0.96\ 10^{-4}\ \text{rds}$$

qui reste de l'ordre de grandeur de la précision requise.

b - Codage des longueurs

On entend par longueur les coordonnées cartésiennes d'un point dans l'espace (poignet, pince...) ainsi que les résultats des calculs effectués sur ces coordonnées (norme,....).

Dans les calculs de transformation géométrique du robot V80, et si on appelle (XP, YP, ZP) la position du poignet (point de concourrance des 3 derniers axes), on aura :

Xp max = 2, Xp min = 0

Yp max = 2, Yp min = -2

Zp max = 2, Zp min = -2

en considérant comme origine le point $O_1$ (intersection axes 1 et 2). Ces valeurs sont théoriques et non physiques.

De même, la valeur maxi de $\sqrt{Xp^2+Yp^2+Zp^2} \leq 2$, les bras 2 et 3 ayant chacun une longueur de 1 m.

La valeur maxi d'une grandeur assimilée à une longueur sera donc de l'ordre de 4 (coordonnée élevée au carré).

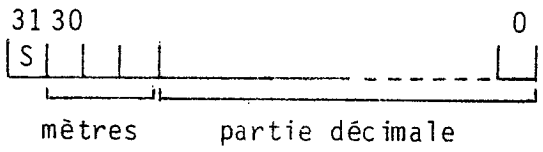
En adoptant le codage classique, la résolution minimale serait :

$$\frac{4}{32768} = 1.2\ 10^{-4}\ m$$

Bien que de l'ordre de grandeur de la précision requise ($10^{-4}$ m), cette résolution apparaît comme insuffisante dans le cas de la commande d'un point autre que le poignet, ou en cas de translation du repère origine.

C'est pourquoi une représentation interne sur 32 bits de longueurs a été adoptée. Par ailleurs, le faible volume des calculs à effectuer sur ces grandeurs rend ce compromis peu pénalisant sur le plan des temps d'exécution.

La représentation choisie est :



où : bit 31 = signe

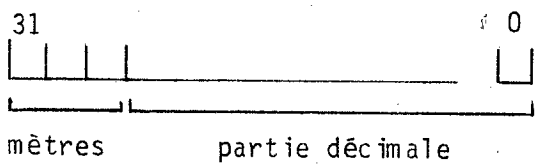
bits 30 à 28 = nombre de mètres

bits 27 à 0 = partie décimale

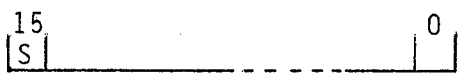
La résolution sur les distances est :

$$\frac{1}{2^{28}} = 3.7\ 10^{-9}\ m$$

Une représentation analogue mais sans bit de signe, a été choisie pour représenter le carré d'une distance.




Enfin, une racine carrée est représentée sur 16 bits selon le format suivant :




c - Codage des fonctions trigonométriques

Ce sont des valeurs variant entre [-1,+1] pour les fonctions

trigonométriques directes et [ 0,2 Π] pour leur inverse.

Ces valeurs sont représentées sur 16 bits en complément à 2, de telle sorte que :

-1 = $8000

+1 = $7FFF

la valeur +1 n'étant pas représentable.

d - Résumé

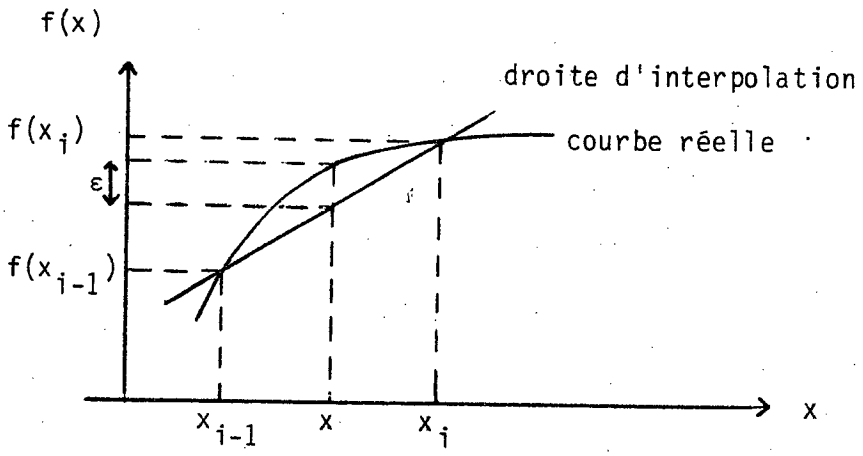
| Type de variable | Exemples | Nombre de | Représentation | Domaine | Résolution |
|---|---|---|---|---|---|
| longueur | Xp, Yp, Zp | 32 | 31 mètres partie décimale 0<br>S x x x x ... x x | [-8,+8[ mètres | $3.7\ 10^{-9}$ m |
| longueur au carré | $X_p^2$, $Y_p^2$, $Z_p^2$ | 32 | 31 mètres partie décimale 0<br>x x x x x ... x x | [0,4] | |
| norme | $X_p^2 + Y_p^2 + Z_p^2$ | 16 | 15 0<br>0 x ... x | [0,2[ mètres | |
| point dans repère pince | XT/6, YT/6, ZT/6 | 16 | 15 0<br>S x x ... x | [-3.27,+3.27[mètres | $10^{-4}$ m |
| angles | $q_1$, $q_2$, $q_3$ ... | 16 | 15 0<br>S x x ... x | [-Π,Π[ | $5.5\ 10^{-3}$ d° |
| valeur fonction trigo | $C_1, S_1, C_2, S_2$ ... | 16 | 15<br>S x x ... x | [-1,1[ | ($9.5\ 10^{-5}$ rds) |

Résumé des différents codages de variables

### 3.3.4. Fonctions mathématiques

#### a - Interpolation linéaire

A partir d'une fonction tabulée, l'interpolation linéaire admet que sur un intervalle séparant 2 valeurs consécutives de la table, l'accroissement de la fonction est proportionnel à celui de l'argument.




$\alpha$ = pas constant de la table (écart entre 2 arguments)

$x$ = argument $\in [x_{i-1}, x_i]$

$\hat{f}(x)$ = valeur interpolée de la fonction

$f(x)$ = valeur réelle de la fonction

$$\hat{f}(x) = f(x_{i-1}) + (x-x_{i-1}) \times \frac{f(x_i)-f(x_{i-1})}{x_i-x_{i-1}}$$

$$\boxed{\hat{f}(x) = f(x_{i-1}) + (x-x_{i-1}) \times \frac{f(x_i)-f(x_{i-1})}{\alpha}} \qquad (1)$$

En pratique, $\hat{f}(x)$ est obtenu en 2 temps :

. Calcul du point d'entrée i dans la table

$$i = \frac{x-x_0}{\alpha}$$

qui nécessite une soustraction et une division.

. Interpolation linéaire, soit 2 accès à la table, 1 addition, 3 soustractions, 1 multiplication et 1 division (formule (1)).

Sur l'intervalle $[x_{i-1}, x_i]$, l'erreur d'interpolation est donnée par :

$\varepsilon(x) = f(x) - \hat{f}(x)$

et $\underset{\max}{\varepsilon(x)}$ est obtenu pour $f'(x) = \hat{f}'(x)$ ( $\varepsilon'(x) = 0$).

Pour déterminer le point de la fonction au voisinage duquel on aura le plus grand écart entre $f(x)$ et $\hat{f}(x)$, il faut considérer la dérivée seconde de la fonction (point de courbure maximale)

$$\boxed{\varepsilon_{max} \text{ en } x_m \Longleftrightarrow f^{(2)}(x_m) \text{ maximum} \Longleftrightarrow f^{(3)}(x_m) = 0} \qquad (2)$$

En considérant un intervalle centré sur $x_m$, on peut écrire :

$$\hat{f}(x_m) = f(x_m - \alpha/2) + \frac{f(x_m+\alpha/2) - f(x_m-\alpha/2)}{\alpha} \times \frac{\alpha}{2}$$

$$= \frac{(f(x_m+\alpha/2) + f(x_m-\alpha/2))}{2}$$

d'où

$$\boxed{\underset{\max}{\varepsilon(X_m)} = f(x_m) - \frac{f(x_m+\alpha/2) + f(x_m-\alpha/2)}{2}} \qquad (3)$$

Cette équation donne, une fois l'argument $x_m$ déterminé par (2), le pas de la table respectant l' $\varepsilon_{max}$ fourni par le cahier des charges. Cette valeur doit être choisie inférieure au pas de discrétisation de la fonction qui dépend directement du codage adopté.

b - Optimisation de l'interpolation

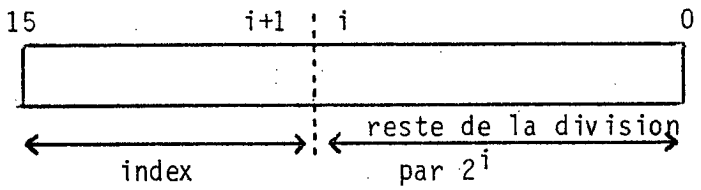
Comme il a été décrit précédemment, $\hat{f}(x)$ s'obtient en 2 temps :

. calcul du point d'entrée

. interpolation

En choisissant un pas de la table égal à une puissance de 2, il est possible de réduire sensiblement le temps d'obtention d'une valeur, ceci en diminuant le nombre de multiplications et divisions à réaliser.

Soit $\alpha = 2^i$. L'argument de la fonction étant codé sur 16 bits, on obtient directement l'index dans la table et le reste de la division par $2^i$ :




Le reste de la division par $2^i$ est obtenu en masquant les bits i+1 à 15, et le quotient en réalisant i décalages à droite.

L'algorithme d'interpolation devient :

| Opérations | Actions |
|---|---|
| 1 soustraction | recaler l'origine $(x-x_0)$, si $x_0 \neq 0$ |
| 1 masque | masquer bits i+1 à 15 $\longrightarrow$ = reste de la division par $2^i$ (=r) |
| i décalages | i décalages à droite $\longrightarrow$ = quotient (q) |
| 2 accès mémoire | accèder à f(q) et f(q+1) |
| 1 soustraction | calculer r = (f(q)-f(q+1))* r/pas |
| 1 multiplication | |
| i décalages | |
| 1 addition | f(x) = f(q) + r |

c - Arctangente

Compte-tenu des calculs à mener, on choisit de calculer un Arctangente à 2 arguments. La fonction n'est tabulée qu'entre $[\ 0, \frac{\Pi}{4}\ ]$, tout calcul étant ramené à cet intervalle, selon le signe des arguments.

Si on applique les calculs de l'erreur d'interpolation, on trouve :

$$f^{(2)}(x) = \frac{-2x}{(1+x^2)^2} \text{ et } f^{(3)}(x) = -2(1+x^2)^2 + 8x^2(1+x^2)$$

$$f^{(3)}(x) = 0 \Longrightarrow x = \pm 1$$

D'où $\varepsilon_{max}$ (par développement en série de Arctg x au voisinage de $\Pi/4$) :

$$|\varepsilon_{max}| = \left| \frac{\Pi}{4} - \frac{\text{Arctg}(1+\alpha/2)+\text{Arctg}(1-\alpha/2)}{2} \right| \simeq \frac{\alpha^2}{8}$$

et le pas de la table :

$$\alpha = \sqrt{8\ |\varepsilon_{max}|}$$

étant codé sur 15 bits (- = -32768), on veut comme erreur maximale possible la valeur d'un bit.

Donc :

$$\alpha = \sqrt{\frac{8\,\Pi}{32768}} = \frac{1}{36.1}$$

Pour optimiser le temps de calcul, il convient de prendre un pas de 1/64 (en gardant la même précision). Par conséquent, 16 valeurs suffiront pour l'intervalle $[\,0, \frac{\Pi}{4}\,]$.

L'organigramme de la fonction Arctangente est donné par la figure 23.

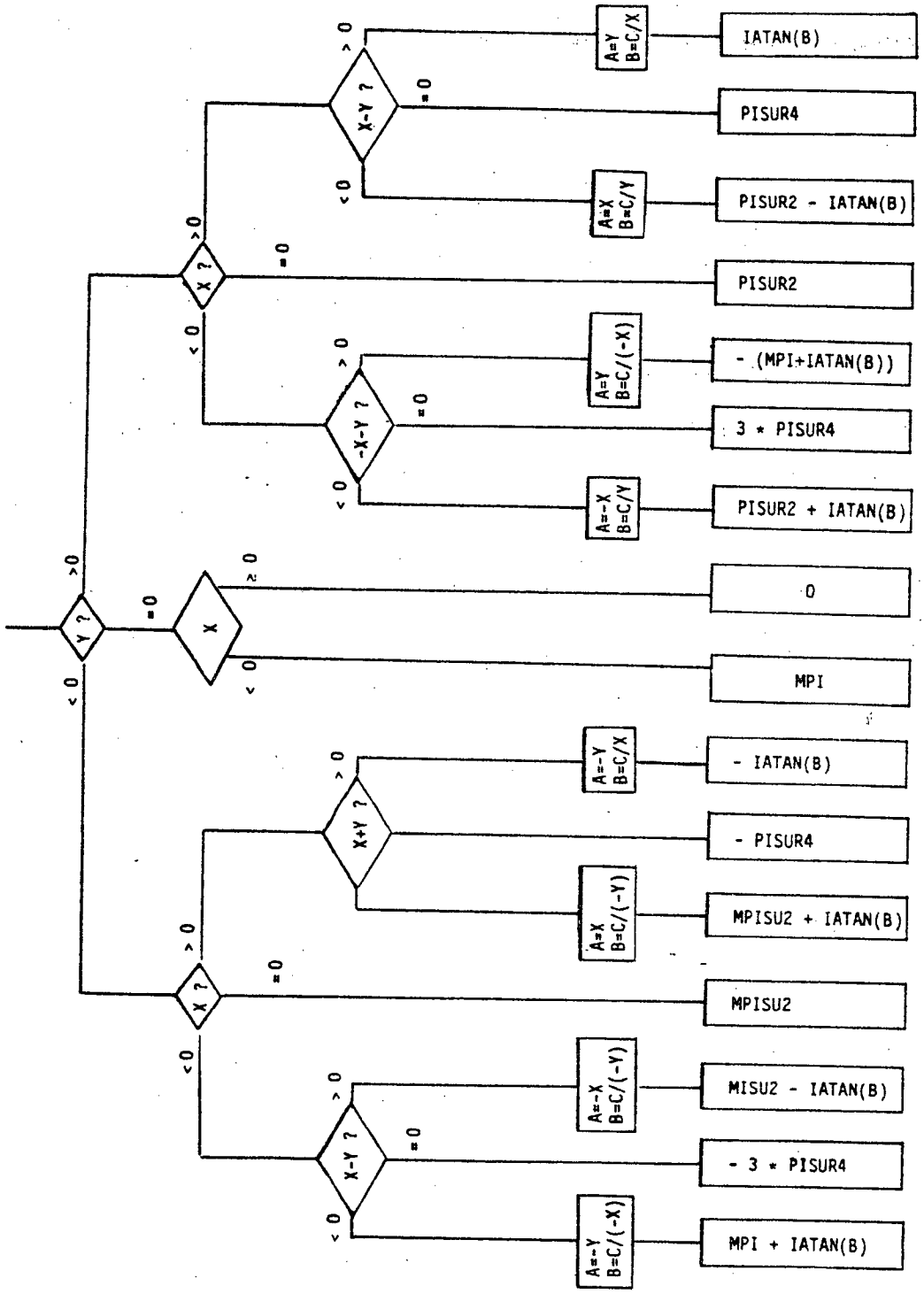



Fig. 23 : Organigramme du calcul d'Arctan (x,y),

avec IATAN fonction tabulée sur

$[0, \frac{\Pi}{4}]$

d - Sinus-Cosinus

Le calcul est similaire à celui d'Arctangente. On a : $f^{(3)}(x) = 0 \Rightarrow x = \frac{\Pi}{2}$, point où l'erreur d'interpolation sera maximale, et vaudra :

$$\varepsilon_{max} = \sin \frac{\Pi}{2} - \sin \frac{(\Pi/2+\alpha/2)+(\sin\Pi/2-\alpha/2)}{2}$$

Développé en série, on obtient :

$$\varepsilon_{max} = 1 - \frac{(1 - \frac{\alpha^2}{4x2})+(1 - \frac{\alpha^2}{4x2})}{2} = - \frac{\alpha^2}{8}$$

Le calcul du sinus (ou du cosinus) d'un angle compris entre $[-\Pi, \Pi[$ se ramène au calcul d'un sinus entre $[0,\frac{\Pi}{2}]$.

$\varepsilon_{max}$ ayant la même valeur que pour le calcul d'Arctangente, une table de 64 valeurs sur $\Pi$ (donc 32 sur $[0,\frac{\Pi}{2}]$) conduira à une précision de $\frac{1}{32768} = 3.10^{-5}$ sur le calcul du sinus ou du cosinus d'un angle compris entre $[-\Pi,\Pi[$ .

L'organigramme de la fonction Sinus est donné par la figure 24, celui de la fonction Cosinus par la figure 25.

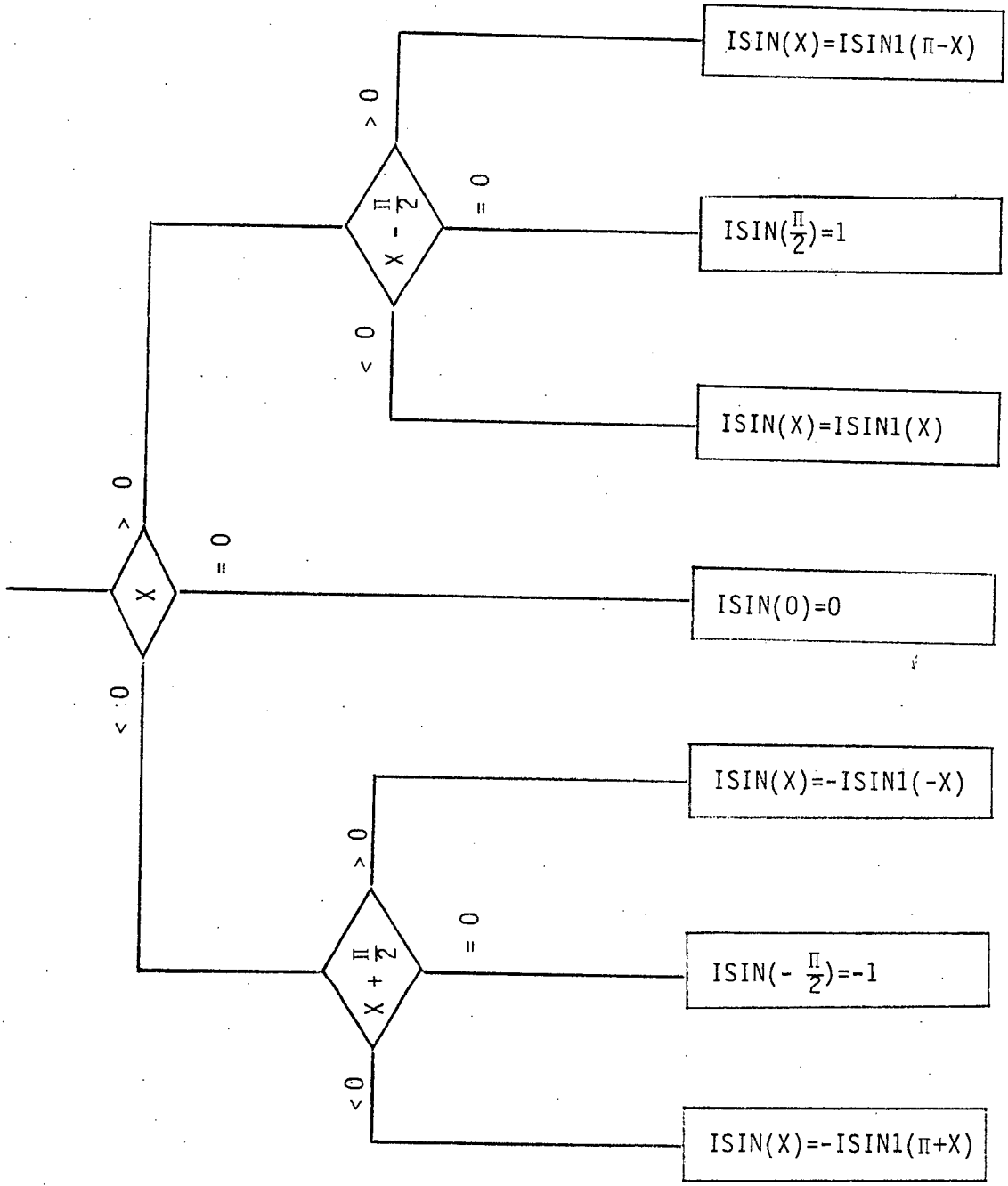



Fig. 24 : Organigramme du calcul de sin(x)

($x \in [-\Pi, \Pi[$) avec ISIN1 fonction

tabulée sur $[0, \frac{\Pi}{2}]$

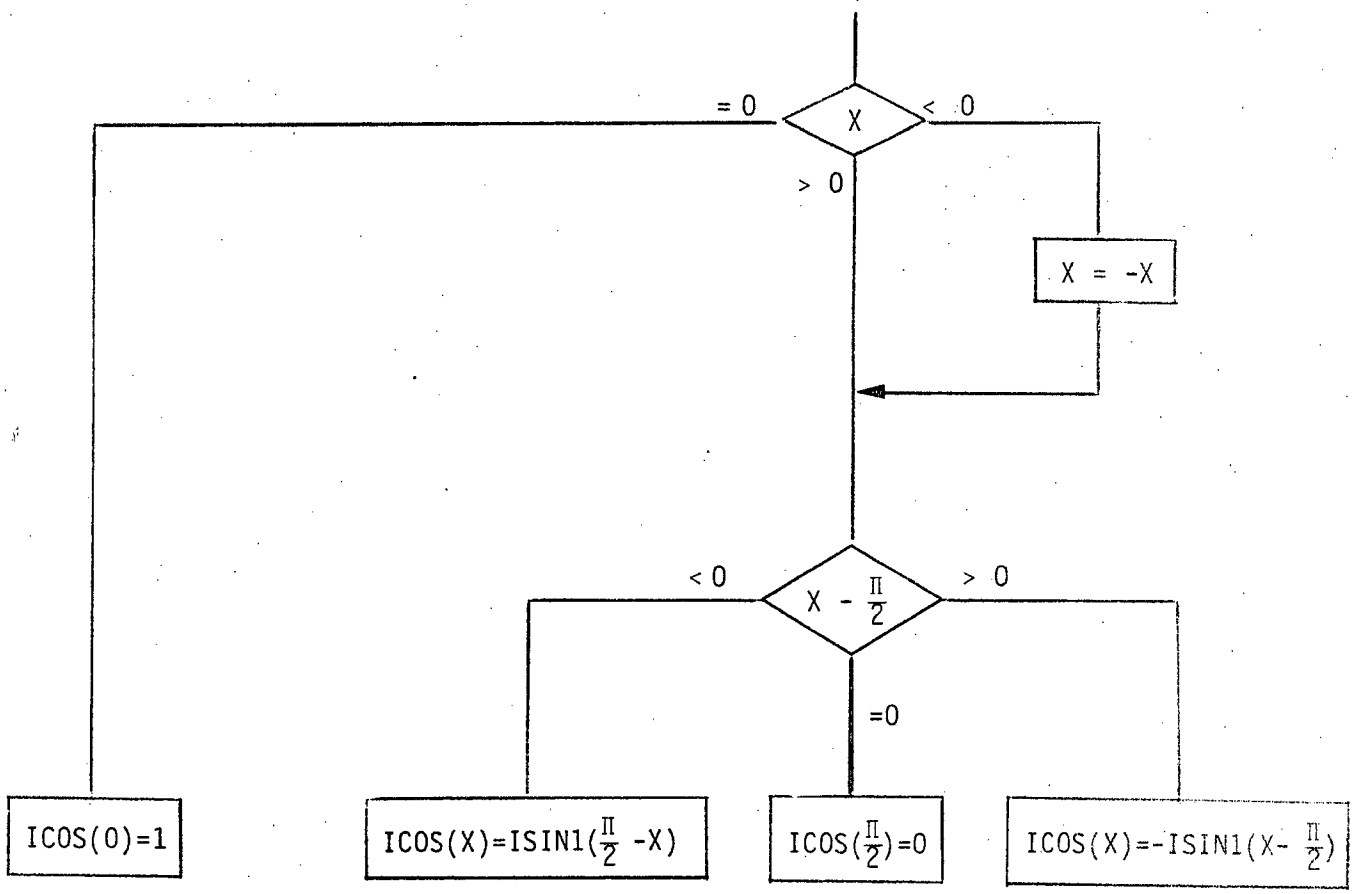



Fig. 25 : Organigramme du calcul de cos(x)

($x \in [-\Pi, \Pi[$)

e - Arcsinus - Arcosinus

$$\text{Arcsin}^{(1)}(x) = (1-x^2)^{-1/2}$$

$$\text{Arcsin}^{(2)}(x) = x(1-x^2)^{-3/2}$$

$$\text{Arcsin}^{(3)}(x) = (1-x^2)^{-3/2}\ [1+3x^2(1-x^2)^{-1}]$$

d'où $\varepsilon_{max}$ pour $x = \pm 1$

L'étude théorique étant moins simple que pour les cas précédents, une simulation en Pascal a conduit à réaliser une tabulation à pas variable de la fonction Arcsinus.

Sa forme est :

Entrée : $[0,1[ = [0,32768]$ sous forme codée

Sortie : $[0,\frac{\Pi}{2}] = [0,16384]$ sous forme codée

| Valeur d'entrée | Point d'entrée de la table | Pas | Nombre de valeurs |
|---|---|---|---|
| 0 à 32000 | TABASIN | 128 | 250 |
| 32001 à 32512 | TABASIN2 | 32 | 16 |
| 32513 à 32752 | TABASIN3 | 8 | 30 |
| 32753 à 32767 | TABASIN4 | 1 | 16 |
| | | | 312 valeurs |

La précision absolue en sortie est d'1 bit, c'est-à-dire $\frac{\Pi}{32768} \simeq 10^{-4}$ radians.

Les algorithmes sont :

IACOS(x)

$x = 0$ IACOS(0) = $\frac{\Pi}{2}$

$x > 0$ IACOS(x) = $\frac{\Pi}{2}$ - IASIN1(x)

$x < 0$ IACOS(x) = $\frac{\Pi}{2}$ + IASIN1(-x)

IASIN(x)

$x = 0$ IASIN(0) = 0

$x > 0$ IASIN(x) = IASIN1(x)

$x < 0$ IASIN = -IASIN1(-x)

avec IASIN1 la fonction Arcsin(x) tabulée sur $[0,\frac{\Pi}{2}]$ .

f - Racine carrée

La méthode de tabulation et d'interpolation linéaire est conservée pour sa facilité de mise en oeuvre et sa rapidité d'exécution.

Dans le cas spécifique du transformateur de coordonnées, les nombres dont on veut extraire la racine carrée sont toujours inférieurs ou égaux à 4.

Afin de limiter la taille des tables, on a choisi de ramener tous les nombres à l'intervalle [1,2[ ou [2,4]. Dès lors, seulement 2 tableaux sont nécessaires :

TABRAC12, contient 128 valeurs qui sont $\sqrt{1+\frac{k}{128}}$, avec k = 0, 1, ... 128

TABRAC24, contient 128 valeurs qui sont $\sqrt{2+\frac{k}{64}}$, avec k = 0, 1, ... 128.

L'interpolation linéaire donnera :

si $x \in [1,2[$ , $\sqrt{x} = \sqrt{1+\frac{k}{128}} + (\sqrt{1+\frac{k+1}{128}} - \sqrt{1+\frac{k}{128}})*$ Reste

si $x \in [2,4]$ , $\sqrt{x} = \sqrt{2+\frac{k'}{64}} + (\sqrt{2+\frac{k'+1}{64}} - \sqrt{2+\frac{k'}{64}})*$ Reste

avec $k = (x-1)* 128$

$k' = (x-2)* 64$

L'accès aux tables pour des nombres plus petits que 1 est réalisé de la manière suivante :

$x \in [0.5,1]$ soit $x_1 = 4x$, on calcule $\sqrt{x_1}$ (TRABAC24)

puis $\sqrt{x} = \frac{1}{2}\sqrt{x_1}$

$x \in [0.25,0.5]$ soit $x_2 = 4x$, on calcule $\sqrt{x_2}$ (TRABRAC12)

puis $\sqrt{x} = \frac{1}{2}\sqrt{x_2}$

Tous les nombres rentrant dans le module racine carrée sont supposés être des longueurs non signées sur 32 bits, tels que les 3 bits de gauche (31-29) représentent le nombre de mètres et les bits 28 à 0 la partie décimale.

Le nombre k est :

K = 28 - n° du premier bit non nul en scrutant le nombre du bit 28 vers le bit 0.

Algorithme :

$k = 2j$ : TRABRAC12, $x' = 2^{2j}$, $\sqrt{x} = \frac{1}{2^j}\sqrt{x'}$

$k = 2j+1$ : TRABRAC24, $x' = 2^{2j}$, $\sqrt{x} = \frac{1}{2^j}\sqrt{x'}$

$k \geq 28$ : Erreur

L'intérêt d'un tel algorithme est que les multiplications et divisions par 2 sont obtenues par des décalages à gauche ou à droite du nombre codé en binaire.

Il est possible d'étendre cet algorithme de calcul de racine carrée à un nombre quelconque codé sur 32 bits : il suffit d'effectuer un transcodage du chiffre d'entrée pour le formater comme le carré d'une distance, en s'arrangeant pour n'avoir que des multiplications ou divisions par des puissances de 2.

g - Résumé des fonctions tabulées

| Fonction | Intervalle d'Entrée | Intervalle de Sortie | Taille minimum nécessaire | Taille implantée * (en mots) |
|---|---|---|---|---|
| SIN | $[0,\frac{\Pi}{2}]$ | $[0,1[$ | 32 | 256 |
| ARCSIN | $[0,1[$ | $[0,\frac{\Pi}{2}]$ | | 312 |
| ARCTAN | $[0,1[$ | $[0,\frac{\Pi}{4}]$ | 16 | 256 |
| RACINE CARREE | $[0,4[$ | $[0,2]$ | | 2x128 |

* La taille effectivement implantée dans les premières versions du système d'exploitation sera réduite (à la taille minimum nécessaire) dans une version ultérieure optimisée.

### 3.3.5. Programmation du modèle géométrique

Les formules et fonctions décrites dans les paragraphes précédents ont été directement programmées en langage Assembleur.

L'usage de macro-instructions a été généralisé afin d'alléger la frappe des programmes, tout en systématisant les procédures de mise au point.

Le programme transformateur de coordonnées est composé d'un module principal (TDC) à 2 points d'entrée (modèle direct, modèle inverse). Toutes les variables de travail sont définies dans ce module. Ce sont :

XT, YT, ZT, $\psi$ , $\theta$ , $\phi$ : Coordonnées opérationnelles du point T

XT_6, YT_6, ZT_6 : Description du point T dans un repère lié au poignet

XP, YP, ZP : Coordonnées opérationnelles du poignet

Q1, Q2, Q3, Q4, Q5, Q6 : Angles mécaniques (coordonnées généralisées)

Q1P, Q2P, Q3P, Q4P, Q5P, Q6P : Angles mécaniques calculés à l'appel précédent

C1, S1 ..... C6, S6 : Lignes trigonométriques des angles mécaniques

CPSI ..... SPHI : Lignes trigonométriques des angles $\psi$ , $\theta$ , $\phi$

SA11 ..... SA33 : Termes de la matrice d'attitude.

Ces variables, définies comme globales, c'est-à-dire accessibles par n'importe quel autre module, sont réactualisées lors de chaque exécution du transformateur. Le conditionnement initial des variables dépend du sens de la transformation. Le modèle direct, par exemple, initialisera Q1 à Q6 en entrée et fournira les positions cartésiennes XT, YT, ZT, $\psi$, $\theta$, $\phi$ en sortie.

Les fonctions réalisées par ce module sont :

- récupération des adresses des paramètres dans la pile
- initialisation des variables
- calculs (appel à des procédures extérieures)
- conditionnement du compte-rendu
- rangement des résultats dans la zone de données de l'appelant
- retour.

L'ordre des opérations est identique pour le transformateur direct (TDCD) ou inverse (TDCI) : seules les procédures extérieures changent.

Les sous-programmes utilisés sont :

SCPTP : calcul des lignes trigonométriques des angles $\psi$ , $\theta$ et $\phi$

CASAPTP : calcul de la matrice d'attitude (SA) par $\psi$ , $\theta$ , $\phi$

XPYPZP : calcul de XP, YP et ZP à partir de XT, YT et ZT

Q1Q2Q3 : calcul de $q_1$, $q_2$ et $q_3$ à partir de XP, YP, ZP

SCQ1Q2Q3 : calcul des Sinus et Cosinus $q_1$, $q_2$, $Q_3$

Q4Q5Q6 : calcul de $q_4$, $q_5$, $q_6$ à partir de $q_1$, $q_2$, $q_3$ et SA. Choix entre 2 solutions

CLTRIGO : calcul des lignes trigonométriques des angles mécaniques

CALCSA : calcul de la matrice d'attitude (SA) à partir des angles mécaniques $q_i$

CXPYPZP : calcul des coordonnées cartésiennes du poignet à partir des $q_i$

XTYTZT : calcul des coordonnées cartésiennes du point T à partir de la position du poignet

CALPTP : calcul de $\psi$ , $\theta$ et $\phi$

et un utilitaire

PMATVEC : produit d'une matrice par un vecteur (passage des XP ⟷ XT).

Les temps d'exécution (mesurés à l'oscilloscope) sont :

Transformation directe : $\simeq$ 2 ms

Transformation inverse : $\simeq$ 3.5 ms

avec une précision de positionnement moyenne égale à $10^{-4}$ ($2.5\ 10^{-4}$ m maximum).

## 3.4. Matrices de couplage tête 3 axes

La chaîne cinématique, entraînant un couplage mécanique sur les 3 derniers axes (décrite en 1-10), doit être prise en compte.

L'optimisation du temps de calcul a conduit à coder les termes des matrices, afin de limiter le nombre d'opérations.

On a :

$$\begin{pmatrix} m_4 \\ m_5 \\ m_6 \end{pmatrix} = pM^{-1}k \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix} - \begin{pmatrix} m_{40} \\ m_{50} \\ m_{60} \end{pmatrix}$$

avec

$\begin{pmatrix} m_4 \\ m_5 \\ m_6 \end{pmatrix}$ = positions compteurs 4, 5 et 6

p = coefficient de formatage

$M^{-1}$ = matrice de couplage inverse

k = coefficient de codage

$\begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}$ = angles mécaniques 4, 5 et 6

$\begin{pmatrix} m_{40} \\ m_{50} \\ m_{60} \end{pmatrix}$ = valeurs codeurs pour $q_4 = q_5 = q_6 = 0$

On a :

$$K = \frac{100}{16384} = \frac{5^2 2^2}{2^{14}}$$

(100 = nombre d'incréments capteur pour PI/2

16384 = codage interne de $\frac{}{2}$ )

en choisissant $p = \frac{1}{5^2 2^4}$, on simplifie le passage aux positions moteurs :

$$\begin{pmatrix} m_4 \\ m_5 \\ m_6 \end{pmatrix} = \frac{1}{2^{16}} M^{-1} \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix} - \begin{pmatrix} m_{40} \\ m_{50} \\ m_{60} \end{pmatrix}$$

La matrice correspondante est :

$$M^{-1} = \begin{bmatrix} -17424 & 0 & 0 \\ -13596 & -19034 & 0 \\ -3168 & -6336 & -17248 \end{bmatrix}$$

En assembleur, le partie haute du résultat du produit de la matrice par le vecteur des angles mécaniques sera <u>directement</u> la position

compteur.

Un raisonnement analogue sur la matrice directe conduit à :

$$\begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix} = \frac{M}{8192} \begin{pmatrix} m_4 - m_{40} \\ m_5 - m_{50} \\ m_6 - m_{60} \end{pmatrix}$$

la division par 8192 étant obtenue par 13 décalages à droite.

$$M = \begin{bmatrix} -30812 & 0 & 0 \\ 22009 & -28205 & 0 \\ 13744 & -10361 & -31127 \end{bmatrix}$$

3.5 Mot d'état du manipulateur

Un mot de 16 bits (FLAG) permet à tout le système de connaître l'état actuel du manipulateur.

La signification des bits est la suivante :

(0 = OFF, 1 = ON)

bit 0 : moniteur en fonction

bit 1 : asservissement en fonction

bit 2 : boîte à bouton en fonction

bit 3 : manipulateur en déplacement (suivi de vitesse ...)

bit 4 : boîte à bouton en fonction (mode cartésien)

bit 7 : témoin d'initialisation

bit 8 : manipulateur sous contrôle du port 2

bit 9 : 0 = vitesse normale, 1 = vitesse lente

bit 10 : origine modifiée

## 3.6. Sécurités - Messages d'erreurs

Les sécurités Hard (Chien De Garde, Sécurité Soft) sont gérées par la tâche asservissement.

Avant chaque début d'exécution, toute commande moniteur doit tester le mot d'état du robot pour savoir si toutes les conditions sont réunies en vue de la bonne exécution :

La gestion de l'espace de travail physique est assurée à 2 niveaux :

- dans la transformation de coordonnées, lors de la définition d'objectifs cartésiens
- dans le module asservissement pour les butées soft de protection (= butée mécaniques - $\varepsilon$).

Dans tous les cas, l'affichage d'un message d'erreur ne peut avoir lieu que dans la tâche moniteur.

Les échanges sont hiérarchisés, chaque tâche ne recevant des commandes que du niveau supérieur auquel elle est sensée renvoyer un compte-rendu (fig. 26).

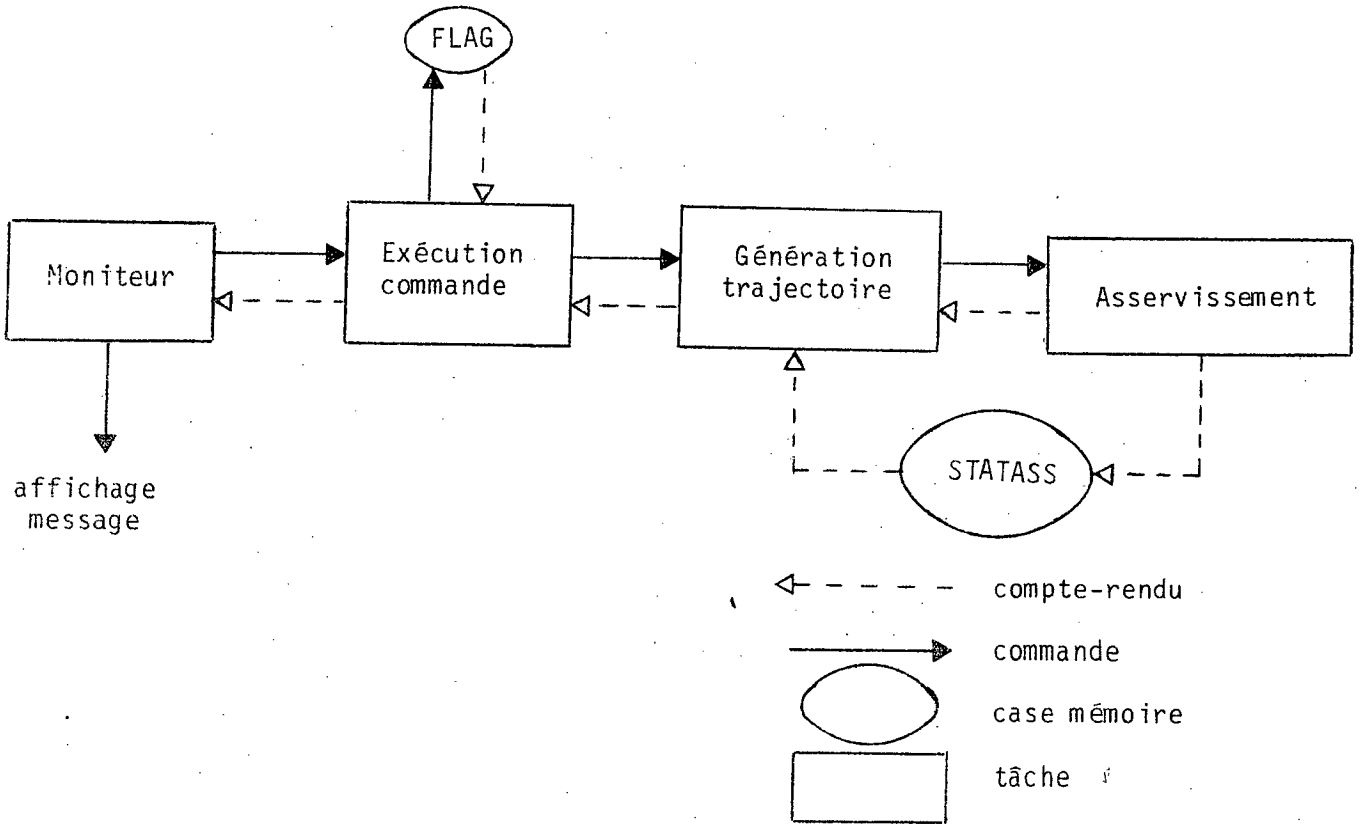



Fig. 26 : Cheminement des commandes et des compte-rendus (seule la tâche moniteur est autorisée à afficher des messages d'erreur)

Quelle que soit la tâche, le compte-rendu passe par le registre DO.L. A chaque retour, l'appelant teste DO.L. S'il est nul, tout s'est bien passé. Sinon, DO.L est renvoyé sans modification au niveau supérieur, et est affiché lorsqu'il arrive au moniteur.

Tout message d'erreur est donc représenté sur 4 caractères, chacun traduisant une erreur spécifique.

Les principaux messages sont :

Module exécution de commande

SEQ. : erreur de séquence. Les conditions actuelles (données par le mot d'état du robot) sont incompatibles avec l'exécution de la commande.

VIT. : erreur vitesse (commande GOx)

VITI : erreur vitesse (commande GOD)

TEMP : erreur temps (commande GOD)

PARi : erreur paramètre i (commandes à paramètres)

DIST : la trajectoire à suivre traverse des butées mécaniques (commande GOD)

Module génération de trajectoire

LIMD : limite en position dépassée sur un des axes (pas de solution dans la transformation de coordonnées inverse)

SOLi : solution i retenue (transformation de coordonnées inverse, i [ 1,8 ] en théorie, [ 1,2 ] en pratique)

Module asservissement :

Compte-tenu de son activation indépendante, la tâche d'asservissement a son propre mot d'état que le module génération de trajectoire peut ou non utiliser.

STATASS est un octet, défini comme étant accessible par toutes les procédures, et dont le bit i est positionné à 1 si l'axe i+1 a atteint sa limite de position. Dans ce cas, la commande reste saturée à la valeur maxi définie dans le fichier des paramètres du manipulateur (CSTE).

## 4. Méthodologie de développement soft

Tout le logiciel présenté a été développé en Assembleur sur le système EXORMACS.

Tous les programmes mis au point sont rangés dans des bibliothèques accessibles par l'ensemble des utilisateurs.

Le développement d'une application nécessite de :

- configurer le fichier TABCOM (commandes moniteur sounhaitées)
- ajouter dans le fichier de commande LNKV.CF les sous-programmes spécifiques assemblés séparement
- lancer une édition de liens par le fichier de commande LNKV.CF.

La sortie est un fichier MONIT.MX directement chargeable et exécutable sur la carte KVM 68000.

Les bibliothèques actuelles sont :

COMMONI : Commandes moniteur
PRIMLM : Primitives LM implantées
COMMAND : Sous-programmes de commande des déplacements du manipulateur
ASSERV : Tâches d'asservissement
TRANSF : Transformateur de coordonnées V80 et sous-programmes associés
TESTROB : Programmes de tests (orientés dépannage matériel essentiellement)
INIT : Sous-programmes d'initialisation, système informatique et manipulateur
DIVERS : Sous-programmes divers de gestion du manipulateur
FCT : Fonctions mathématiques et tables associées
UTILIT : Utilitaires système (essentiellement traducteurs)
SYSTEM : PARAM = paramètres de l'application (gains...)
CSTE = constantes du manipulateur.
ADRS = adresses physiques (E/S, compteurs ...)

L'image mémoire créée est partionnée en 15 sections réparties sur 3 segments, ceci en vue d'une implantation future en ROM et d'un couplage à des modules Pascal.

La décomposition retenue est donné par la fig. 27.

| SEGMENT | SECTION | CONTENU |
|---|---|---|
| 0 | 0 | Ram utilisateur + robot |
| | 1 | Source système robot de base |
| | 2 | Source moniteur + TABCOM |
| | 3 | Commandes moniteur |
| | 4 | Tableaux + constantes (données) |
| | 5 | Source transformateur de coordonnées |
| 1 | 6 | Sous-programmes LM |
| | 9 | Sous-programmes compatibles PASCAL |
| | 10 | Programmes utilisateurs (chargés en overlay à partir de l'EXORMACS) |
| | 11 | Réservé soft en cours de développement |
| 2 | 12 | Ram à initialiser |
| | 14 | Ram disponible (programmes utilisateurs) |
| | 15 | Ram des programmes Pascal |

Fig. 27 : Décomposition en 3 segments et 16 sections d'une application

Le fichier de commande permettant la génération d'une image mémoire se présente sous la forme suivante (LNKV.CF) :

```
=LINK ,,LIST.SA;QAMXUISZ=64                    Ligne de commande
SEG SEG0(UG):2 $1C00                           Partitionnement mémoire
SEG SEG1(UG):0,1,3-15
IN MONIT,TABCOM                                Moniteur+fichier des commandes
IN DROIT1,DROIT2,SPSUIV,DEMO,LIGD,PARAM        Programmes hors bibliothèques
IN MANUEL,PGASPO,CSTE,CONTVER,DEPCART
IN DOIGTS
LIB 87.SUBV80.COMMONI.RO                       Scrutation bibliothèques
LIB 87.SUBV80.PRIMLM.RO
LIB 87.SUBV80.COMMAND.RO
```

```
LIB 87.SUBV80.ASSERV.RO
LIB 87.SUBV80.TRANSF.RO
LIB 87.SUBV80.TESTROB.RO
LIB 87.SUBV80.INIT.RO
LIB 87.SUBV80.DIVERS.RO
LIB 87.SUBV80.FCT.RO
LIB 87.SUBV80.UTILIT.RO
LIB 87.SUBV80.SYSTEM.RO
LISTU
END
=END
```

] Scrutation bibliothèques

] Liste références non résolues

Les programmes d'entrées/sorties (DRIVER) sont générés et chargés séparément. Le fichier de commande utilisé se nomme LNKD.CF :

```
=LINK DRIVER;QAMXUIS
START 0 $1000
START 1 $1A00
START 2 $1800
LISTU
END
=END
```

] Ligne de commande

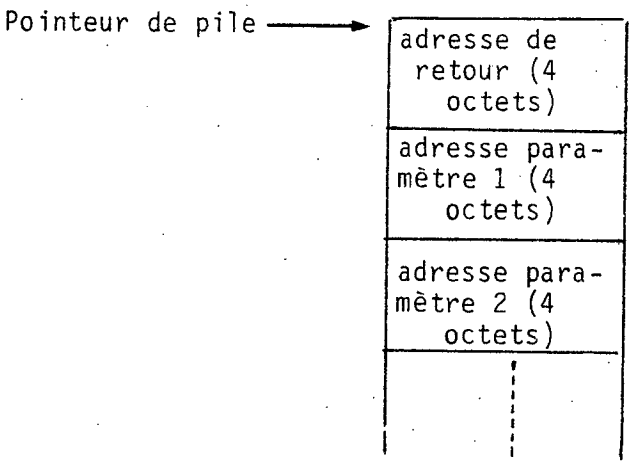
] Adresses d'implantation physique

Une fois un programme mis au point, il est intégré dans la bibliothèque correspondante.

En règle générale, on s'arrange pour assurer un passage de paramètres entre procédures qui soit compatible PASCAL : on facilite ainsi l'interface futur entre des programmes écrits dans des langages différents.

Tous les paramètres peuvent être passés par la pile, si possible par adresse et non par valeur.

Après l'appel du sous-programme, la pile à la forme : < 10 >



Dans certains cas, l'adresse du paramètre peut-être remplacé pour sa valeur formelle.

Au retour, le haut de la pile contient la ou les valeurs renvoyées par la fonction ou procédure.

Bien que nécessitant une gestion de la pile, ce procédé une fois étendu permet de générer des applications contenant des modules écrits en PASCAL, en FORTRAN et Assembleur.

Le respect de ces différentes contraintes permettra une évolution permanente et contrôlée de l'environnement de programmation, sans avoir à remettre perpétuellement en cause l'acquis.

## 5. Conclusion

Une fois le système de base opérationnel, les efforts vont maintenant porter essentiellement sur :

- l'amélioration de la commande
- l'augmentation de la puissance de calcul, avec l'implantation d'une 2$^{ème}$ unité centrale 68000
- la programmation par un langage de haut niveau (PASCAL, FORTRAN ...) de certains algorithmes, notamment la génération et le contrôle de trajectoire.

## BIBLIOGRAPHIE

(1) Dépannage sur site des robots Renault - Acma type 80, partie électrique,
ACMA-ROBOTIQUE.

(2) M. FOGEL, H. JAMET

Participation à un projet de robotique, Rapport d'option, INRIA, Juin 1982.

(3) Versabug debugging package user's guide

Motorola, Mai 1981.

(4) J.F. MIRIBEL, E. MAZER

Manuel de référence LM

IMAG, Octobre 1982.

(5) K.G. SHIN, S.B. MALIN

Dynamic adaptation of robot kinematic control to its actual behavior

Robotic Laboratory, Rensselaer Polytechnic Institute, 1981.

(6) V. FUERTES

Logiciel de commande du robot manipulateur RV80

Note interne DERA 251/82, Décembre 1982.

(7) P. MOREAU, J.P. SANTAMARIA

Etude de trajectoires pour un manipulateur V80 Acma-Cribier

DEA d'Informatique, Université Paris 11, Septembre 1983.

(8) G. DENIAUD

Commande du robot V80

DEA d'Automatique, Université Paul Sabatier, Septembre 1983.

(9) R. MAMPEY, M. LLIBRE

Etude comparative sur la commande des robots industriels

Rapport DERA, Juin 1981.

(10) M68000 Resident Pascal User's Manual

4th edition, Motorola, 1982.

Imprimé en France

par

l'Institut National de Recherche en Informatique et en Automatique