REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

»o«

MINISTERE AUX UNIVERSITES

»o«

ECOLE NATIONALE POLYTECHNIQUE

»o«

THESE

Pour l'obtention du diplôme de

MAGISTER

EN GENIE INDUSTRIEL

ETUDE ET REALISATION D'UN MULTIPLEXEUR STATISTIQUE HUIT VOIES PAR LE BIAIS DE LA SIMULATION

présentée par :

Mme TEBIBEL née BOUABANA Thouraya

ingénieur d'état en informatique

soutenue le 18 avril 1992 devant la commission d'examen :

M. Z. HADDAD ..... Président
M. L. KERBACHE ..... Rapporteur
M. R. TOUMI ..... Examinateur
Mme H. BENCHERIF ..... Examinatrice
M. T. LAMRAOUI ..... Examinateur
M. B. SANSAL ..... Invité

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

»o«

MINISTERE AUX UNIVERSITES

»o«

ECOLE NATIONALE POLYTECHNIQUE

»o«

THESE

Pour l'obtention du diplôme de

MAGISTER

EN GENIE INDUSTRIEL

ETUDE ET REALISATION D'UN MULTIPLEXEUR STATISTIQUE HUIT VOIES PAR LE BIAIS DE LA SIMULATION

présentée par :

Mme TEBIBEL née BOUABANA Thouraya

ingénieur d'état en informatique

soutenue le 18 avril 1992 devant la commission d'examen :

M. Z. HADDAD ..... Président
M. L. KERBACHE ..... Rapporteur
M. R. TOUMI ..... Examinateur
Mme H. BENCHERIF ..... Examinatrice
M. T. LAMRAOUI ..... Examinateur
M. B. SANSAL ..... Invité

RESUME

L'essor du matériel informatique accompagné du besoin toujours croissant de communiquer à distance, a rendu la connexion directe de tout ce matériel très coûteuse. L'apparition des multiplexeurs statistiques, objet de notre thèse, a permis d'apporter une solution satisfaisante à ce problème en établissant la communication entre plusieurs points de deux sites éloignés à travers une liaison unique. Notre produit subira, avant sa réalisation pratique, une étude par simulation qui déterminera le comportement de ses variables, les valeurs de ses paramètres critiques, et étudiera ses limites en vue d'éventuelles optimisations et extensions. Notre simulation ne portera que sur le côté logiciel du projet, le côté matériel ne nécessitant point de simulation en raison des énormes progrès technologiques de la dernière décennie.

The boom in data processing equipments, along with the increasing needs to communicate from a distance, has made the direct connection of all this equipment very expensive. The invention of the statistical multiplexers, topic of our thesis, brought a satisfying solution to this problem by establishing communication between many points of two distant sites using only one telephone line. Before its practical accomplishment, our product will go through a study by simulation determining the reaction of its variables, the value of its critical parameters and studying its limits for purpose of eventual optimizations and extensions. This simulation concerns only the software side of the project. The hardware does not entail simulation because of the great technological progress in this last decade.

إن تطور آلات الإعلام الآلي مصحوبا بحاجة دائمة الإزدياد للإتصال عن بعد جعل الإرتباط المباشر لكل هته الآلات جد مكلفا. وظهور المجمع الإحصائي، موضوع دراستنا قد سمح بإيجاد حل مرض لهذا المشكل، وذلك بإحداث إتصال لعدة نقاط لموقعين متباعدين عن طريق رابطة واحدة. سوف يتعرض منتوجنا، قبل الإنجاز التطبيقي لدراسة تظاهرية تحدد تصرف متغيراته، وقيمه الوسيطية الحرجة، و سوف ندرس حدوده بغية التجويدات و التوسيعات المحتملة. تظاهرنا هذا لن يعتني إلا بالجانب البرنامجي للمشروع فالجانب المادي لا يقتضي التظاهر البتة بسبب التطورات التكنولوجية الضخمة للعشرية الأخيرة.

A V A N T - P R O P O S

Au terme de la rédaction de ce mémoire, il m'est de tâche agréable de remercier toutes les personnes qui ont contribué à son élaboration.

Ce travail a été réalisé au sein de l'Entreprise Nationale des Systèmes Informatiques (ENSI). Que Monsieur N. BERRAH, ex-Directeur Général de l'ENSI et Monsieur T. DILMI, ex-Directeur de la Recherche et du développement (DRD) de l'ENSI soient remerciés pour avoir autorisé, financé et encouragé le projet.

Ma profonde gratitude va à Monsieur L. KERBACHE, Professeur à l'école GROUPE ESC de Rennes (France), qui a assuré la direction scientifique de cette thèse. Qu'il soit remercié pour ses conseils et avis éclairés, ainsi que pour la confiance qu'il m'a toujours témoignée.

Je voudrais remercier le Docteur Z. HADDAD, pour ses critiques et ses conseils fructueux et pour m'avoir fait l'honneur de présider le jury. Je fait également, part de mes remerciements au Professeur R. TOUMI, au Docteur H. BENCHRIF et au Docteur T. LAMRAOUI pour avoir accepté de juger ce travail et participer au jury. Que le Professeur B. SANSAL trouve ici toute ma gratitude pour avoir bien voulu répondre à notre invitation et honorer de sa présence le jury de soutenance.

Le Docteur H. DAMERDJI a suivi mon travail avec un grand intérêt. Qu'il soit assuré de ma reconnaissance pour le temps qu'il m'a consacré et pour les critiques constructives qu'il m'a prodiguées.

Que Monsieur N. BADACHE et Madame W. ABDERRAHIM chargés de cours à l'institut d'informatique de l'U.S.T.H.B. soient remerciés pour leurs critiques et leurs conseils pertinents.

Je remercie enfin, Monsieur A. TEBIBEL, Directeur de Projet "Transmission de Données" à la DRD de l'ENSI pour avoir suivi le projet et aidé à son accomplissement.

S O M M A I R E

INTRODUCTION.....1

CHAPITRE 1.- GENERALITES SUR LES MULTIPLEXEURS ET CONCENTRATEURS
1.- Introduction.....3
2.- Les multiplexeurs.....3
3.- Les concentrateurs.....14
4.- Protocole de communication SDLC.....15
5.- Principales caractéristiques techniques des multiplexeurs statistiques.....16

CHAPITRE 2.- DEVELOPPEMENTS THEORIQUES
1.- Modèle de multiplexage.....18
2.- Modèle de démultiplexage.....22
3.- Modèle à entrée mixte et service constant.....26
4.- Modèle à priorités multiples.....28
5.- Modèle à entrée binomiale et service constant.....30
6.- Modèle à retransmissions ARQ.....33
7.- Modèle à entrées corrélées.....35
8.- Modèle à tampons séparés.....37
9.- Modèle à tampons réduits.....40
10.- Conclusion.....42

CHAPITRE 3.- SYSTEME MATERIEL.
1.- Descriptif.....43
2.- Microprocesseur.....45
3.- Interface de communication asynchrone octal.....49
4.- Interface de communication synchrone/asynchrone dual.....54
5.- Port d'entrée/sortie.....59
6.- Partitionnement de la mémoire.....59
7.- Emetteurs/récepteurs de ligne.....60
8.- Jonctions RS232.....61

CHAPITRE 4.- PRODUIT SIMULE
1._ Qu'est-ce-que la simulation?.....62
1.1.- Introduction.....62
1.2.- Systèmes et modèles.....64
1.3.- Types de simulation.....65
1.4.- Langages de simulation.....68
1.5.- SLAM II.....71

1.6.- Méthodologie de simulation.....74
2.- Simulation du système.....79
2.1.- Description du système.....79
2.2.- Modélisation du système.....89
2.3.- Collecte et analyse des données.....96
2.4.- Vérification et validation du modèle.....00
2.5.- Analyse des sorties.....106
2.6.- Interprétation des résultats et exploitation du modèle115
2.7.- Apport de la simulation pour la conception pratique...134

CHAPITRE 5.- SYSTEME LOGICIEL
1.- Temps partagé.....135
2.- Parallélisme des processus.....136
3.- Interruptions.....138
4.- Structure de données.....140
5.- Description des processus.....144
6.- Protocole de communication.....148
7.- Synchronisation des processus.....153
8.- Algorithmes.....160

CHAPITRE 6.- CONCLUSIONS ET EXTENSIONS.....170

BIBLIOGRAPHIE.....172

ANNEXES.....178
1.- Annexe A.....178
2.- Annexe B.....180
3.- Annexe C.....182
4.- Annexe D.....184
5.- Annexe E.....186
6.- Annexe F.....188

T A B L E D E S M A T I E R E S

INTRODUCTION.....1

CHAPITRE 1.- GENERALITES SUR LES MULTIPLEXEURS ET CONCENTRATEURS.3
1.- Introduction.....3
2.- Les multiplexeurs.....3
2.1.- Multiplexage statique.....5
2.1.1.- Multiplexage en fréquence.....5
2.1.2.- Multiplexage en temps.....6
2.1.2.1.- Multiplexage temporel par caractère.....6
2.1.2.2.- Multiplexage temporel par bit.....8
2.2.- Multiplexage temporel statistique.....9
2.2.1.- Fonctionnement.....10
2.2.2.- Performances.....12
2.3.- Possibilités des différents types de multiplexeurs.....12
3.- Les concentrateurs.....14
4.- Protocole de communication SDLC.....15
5.- Principales caractéristiques techniques des multiplexeurs statistiques.....16

CHAPITRE 2.- DEVELOPPEMENTS THEORIQUES.....17
1.- Modèle de multiplexage.....18
2.- Modèle de démultiplexage.....22
3.- Modèle à entrée mixte et service constant.....26
4.- Modèle à priorités multiples.....28
5.- Modèle à entrée binomiale et service constant.....30
6.- Modèle à retransmissions ARQ.....33
7.- Modèle à entrées corrélées.....35
8.- Modèle à tampons séparés.....37
9.- Modèle à tampons réduits.....40
10.- Conclusion.....42

CHAPITRE 3.- SYSTEME MATERIEL.....43
1.- Descriptif.....43
2.- Microprocesseur.....45
2.1.- Généralités.....45
2.2.- Organisation matérielle.....45
2.2.1.- Signaux horloges.....47
2.2.2.- Signaux de contrôle.....47
2.3.- Architecture interne.....48

3.- Interface de communication asynchrone octal.....49
3.1.- Contrôle d'interruption.....50
3.2.- Contrôle d'opérations.....52
3.3.- Circuit d'horloge.....52
3.4.- Emission.....53
3.5.- Réception.....53
3.6.- Broches d'entrée à service multiple.....54
3.7.- Caractéristiques générales.....54
4.- Interface de communication synchrone/asynchrone dual.....54
4.1.- Boucle de surveillance.....56
4.2.- Interruptions.....56
4.2.1.- Interruption d'émission.....58
4.2.2.- Interruption de réception.....58
4.2.3.- Interruption de réception spéciale.....58
4.2.4.- Interruption externe.....59
5.- Port d'entrée/sortie.....59
6.- Partitionnement de la mémoire.....59
7.- Emetteurs/récepteurs de ligne.....60
8.- Jonctions RS232.....61

CHAPITRE 4.- PRODUIT SIMULE.....62
1.- Qu'est-ce-que la simulation?.....62
1.1.- Introduction.....62
1.1.1.- Définition de la simulation.....62
1.1.2.- Intérêt de la simulation.....63
1.2.- Systèmes et modèles.....64
1.2.1.- Le système.....64
1.2.2.- Le modèle.....64
1.3.- Types de simulation.....65
1.3.1.- Simulation continue.....65
1.3.2.- Simulation discrète.....66
1.3.3.- Simulation combinée.....67
1.4.- Langages de simulation.....68
1.4.1.- Langages de simulation continue.....69
1.4.1.- Langages de simulation discrète.....69
1.4.1.- Langages de simulation combinée.....70
1.5.- SLAM II.....71
1.5.1.- Présentation générale du langage.....71
1.5.2.- Possibilités offertes par SLAM II.....72
1.5.3.- Structure du programme SLAM II.....73
1.5.3.1.- Le programme principal.....73
1.5.3.2.- Les programmes périphériques.....73
1.5.4.- Instructions de contrôle de la simulatio.....74

1.6.- Méthodologie de simulation.....74
1.6.1.- Formulation du problème.....75
1.6.2.- Modélisation.....75
1.6.3.- Collecte et analyse des données.....76
1.6.4.- Transcription informatique du modèle.....76
1.6.5.- Vérification du modèle.....76
1.6.6.- Validation du modèle.....77
1.6.7.- Stratégie et tactique de simulation.....77
1.6.8.- Exécution de la simulation.....78
1.6.9.- Analyse des sorties.....78
1.6.10.- Interprétation des résultats, exploitation et développements futurs du modèle.....78
2.- Simulation du système.....79
2.1.- Description du système.....79
2.1.1.- Décomposition du système en processus.....80
2.1.2.- Organisation des processus.....82
2.1.2.1.- Contrainte de temps imposée par le transfet d'un caractère.....82
2.1.2.2.- Interruptions sur toutes les voies.....83
2.1.2.3.- Boucle de test sur toutes les voies.....84
2.1.2.4.- Boucle de test sur les voies basse vitesse...86
2.2.- Modélisation du système.....89
2.2.1.- Modélisation des arrivées.....90
2.2.2.- Modélisation du processus gestion du superviseur..91
2.2.3.- Modélisation des processus réception sur une voie basse vitesse.....91
2.2.4.- Modélisation des processus constitution de la trame d'une voie basse vitesse.....92
2.2.5.- Modélisation du processus émission sur la voie haute vitesse.....93
2.2.6.- Modélisation du processus réception sur la voie haute vitesse.....94
2.2.7.- Modélisation des processus émission sur une voie basse vitesse.....95
2.2.8.- Caractéristiques du modèle final.....95
2.3.- Collecte et analyse des données.....96
2.3.1.- Acqusition des services sur avis d'expert.....97
2.3.2.- Acqusition des services sur système réel.....97
2.3.3.- Ajustement d'une loi exponentielle aux inter-arrivées du superviseur.....98
2.4.- Vérification et validation du modèle.....100
2.4.1.- Vérification du modèle.....100
2.4.2.- Validation du modèle.....101

2.4.2.1.- Approche par inspection.....103
2.4.2.2.- Approche par intervalle de confiance.....104
2.5.- Analyse des sorties.....106
2.5.1.- Etat stationnaire/ état transitoire.....107
2.5.2.- Simulation terminale/simulaion en état stationnaire107
2.5.3.- Construction de l'intervalle de confiance.....108
2.5.3.1.- Procédure des répliques.....109
2.5.3.2.- Procédure des «batch means».....109
2.5.3.3.- Procédure des cycles régénératifs.....110
2.5.3.4.- Procédure de la covariance.....111
2.5.4.- Stratégie de départ.....112
2.5.4.1.- Etablissement des cooditions initiales.....112
2.5.4.2.- Procédure de troncature.....113
2.5.5.- Analyse des sorties du modèle.....113
2.6.- Interprétation des résultats et exploitation du modèl.115
2.6.1.- Optimisation de la longueur de la trame.....115
2.6.2.- Constitution simultanée d'une trame sur les huit mémoires tampons de réception.....127
2.6.3.- Plus grand débit des voies basse vitesse égal à celui de la voie composite.....128
2.6.4.- Plus grand débit des voies basse vitesse égal à 4800 bauds.....129
2.6.5.- Taux de concentration supérieur à 100 %.....132
2.7.- Apport de la simulation pour la conception pratique...134

CHAPITRE 5.- SYSTEME LOGICIEL.....135
1.- Temps partagé.....135
2.- Parallélisme des processus.....136
3.- Interruptions.....138
4.- Structure de données.....140
4.1.- Mémoires tampons.....140
4.2.- File d'attente.....142
4.3.- Régulation du flôt des données.....143
5.- Description des processus.....144
5.1.- Gestion du superviseur.....144
5.2.- Réception sur une voie basse vitesse.....146
5.3.- Constitution de la trame d'une voie basse vitesse.....146
5.4.- Emission sur la voie haute vitesse.....147
5.5.- Réception sur la voie haute vitesse.....147
5.6.- Emission sur une voie basse vitesse.....147
6.- Protocole de communication.....148
6.1.- Initialisation.....149
6.2.- Emission SDLC.....150

6.2.1.- Début d'émission.....150
6.2.2.- Emission des données.....150
6.2.3.- Détection d'une fin de trame.....151
6.3.- Réception SDLC.....152
6.3.1.- Réception des données.....152
6.3.2.- Détection d'une fin de trame.....153
7.- Synchronisation des processus.....153
7.1.- Détermination des ressources critiques.....154
7.1.1.- Partage du processeur.....154
7.1.2.- Partage de la mémoire.....154
7.1.3.- Partage de programme.....154
7.2.- Construction des sections critiques.....155
7.2.1.- Mémoires tampons.....156
7.2.2.- Pointeurs de mémoires tampons.....156
7.2.3.- Compteurs.....157
7.2.4.- Registres.....157
7.2.5.- Variables.....157
7.3.- Mécanisme de synchronisation.....158
7.3.1.- Principe.....158
7.3.2.- Mise en oeuvre.....158
8.- Algorithmes.....160
8.1.- Réception d'un caractère sur une voie basse vitesse...160
8.2.- Réception d'un bloc de trois caractères sur une voie basse vitesse.....161
8.3.- Constitution de la trame d'une voie basse vitesse.....162
8.4.- Interruption de la voie composite.....164
8.5.- Emission d'un caractère sur la voie haute vitesse.....166
8.6.- Réception d'un caractère sur la voie haute vitesse....168
8.7.- Emission d'un caractère sur une voie basse vitesse....169

CHAPITRE 6.- CONCLUSIONS ET EXTENSIONS.....170

BIBLIOGRAPHIE.....172

ANNEXES.....178
1.- Annexe A.....178
2.- Annexe B.....180
3.- Annexe C.....182
4.- Annexe D.....184
5.- Annexe E.....186
6.- Annexe F.....188

INTRODUCTION

Les premiers systèmes de télécommunications (télégraphie, téléphone) sont nés au XIX^{e} siècle, mais ce n'est qu'aux environs des années 1980 que la téléinformatique entrait dans une phase de déploiement à grande échelle dans le domaine professionnel. Et, que de chemin parcouru depuis, avec l'avènement de la télématique, la prolifération des terminaux et des micro-ordinateurs connectés, l'évolution spectaculaire des techniques et des produits, les progrès considérables en matière de formalisation et de normalisation des architectures, interfaces et protocoles!

Ainsi, l'essor du matériel informatique accompagné du besoin toujours croissant de communiquer à distance, a rendu la connexion directe de tout ce matériel très coûteuse pour deux raisons:

1- le matériel nécessaire à la connexion d'un terminal sur l'ordinateur central représente un coût non négligeable, et la gestion des transmissions pour un nombre important de terminaux mobilise la puissance de l'ordinateur central pour des tâches auxquelles il n'est pas adapté: fonctions de communication et de transport, et traitements comportant de nombreuses interruptions;
2- les lignes de transmission représentent un coût important dans l'ensemble de l'installation, alors qu'elles ont un faible taux d'utilisation (un terminal conventionnel n'utilise réellement la ligne que pendant quelques pour cents du temps).

Le point 1) fut partiellement résolu par l'apparition des multiplexeurs statiques, malheureusement au détriment de l'ordinateur central qui a vu sa charge accrue par la gestion des liaisons. L'introduction des multiplexeurs statistiques, objet de notre thèse, et des concentrateurs a permis d'apporter une réponse satisfaisante aux points 1) et 2).

Tout projet d'envergure, doit subir, avant sa réalisation pratique, une étude par simulation permettant de déterminer le comportement de ses variables, les valeurs de ses paramètres critiques, et d'étudier ses limites en vue d'éventuelles optimisations et extensions; nous avons par conséquent, jugé opportun de ne point négliger cette phase et d'explorer profondément ses différentes tâches. Cette simulation ne portera que sur le côté logiciel du

projet, le côté matériel ne nécessitant pas de telle procédure, grâce aux énormes progrès technologiques de la dernière décennie.

En effet, l'apparition sur le marché, aux environs des années 1986, de nouveaux composants intégrés (interface de communication octal intégré) et fortement sophistiqués (interface de communication doté de protocole de communication), nous a permis de procéder à de considérables optimisations sur l'architecture de la carte multiplexeur qui, jusque là, était assez complexe et fortement encombrée.

Une fois le matériel construit, nous nous sommes fixés pour tâche de l'habiller du logiciel permettant son fonctionnement. Ce logiciel assurera la communication de plusieurs points de deux sites éloignés, à travers un partage optimal de la voie reliant les deux sites. c'est un logiciel d'application en temps réel caractérisée par la brièveté du temps de réponse, les contraintes de vitesse étant par conséquent, fondamentales.

Le chapitre 1 de ce document, présente les différents types de multiplexeurs existants ainsi que leurs performances et possibilités. Il introduit également les concentrateurs et définit le protocole de communication SDLC.

Le chapitre 2 expose les principales études effectuées sur les multiplexeurs statistiques et énonce les principaux résultats obtenus.

L'architecture de la carte multiplexeur sera exposée dans le chapitre 3.

Le chapitre 4 sera exclusivement consacré à l'étude du produit par simulation. Nous y trouverons un aperçu sur la simulation, suivi d'une étude détaillée du produit simulé et d'une analyse des résultats obtenus.

Dans le chapitre 5, nous découvrirons les différentes fonctions du logiciel et leur mécanisme, ainsi que la structure de ses données.

Enfin, au chapitre 6 une conclusion fera le point sur les différents résultats obtenus et des suggestions destinées à de nouvelles extensions de l'étude seront avancées.

CHAPITRE 1

GENERALITES SUR LES MULTIPLEXEURS ET CONCENTRATEURS

1.- INTRODUCTION

Lorsque plusieurs circuits de données doivent être réalisés en parallèle entre deux sites éloignés, il peut être intéressant de les juxtaposer sur un support téléphonique unique.

Ainsi, l'objectif principal des multiplexeurs est de réduire le coût des lignes en optimisant leur charge: pour M liaisons déportées à établir, l'on fait l'économie de (2M-2) modems et de (M-1) lignes.

Il existe deux types de matériels pour réaliser ce regroupement: les multiplexeurs et les concentrateurs.

2.- LES MULTIPLEXEURS [01, 02, 03, 04]

La structure fonctionnelle d'une liaison utilisant un multiplexeur est la même (figure 1.1) quelle que soit la méthode de multiplexage utilisée:
— un organe de multiplexage, noté MUX ou DEMUX,
— des organes de voies basse vitesse, notés OVBV,
— un organe de voie haute vitesse, noté OVHV.

Les caractéristiques qui permettent de comparer les différentes méthodes de multiplexage sont :
— la rapidité de transfert exprimée en car/s ou bit/s.
— L'efficacité: les voies basse vitesse (BV) ont un débit Ci car/s et chaque caractère contient Ni bits utiles (sans START et STOP). La voie haute vitesse (HV) a un débit D bit/s.
L'efficacité:

$$e = D^{-1} \sum_{i} CiNi$$

doit être la plus grande possible afin d'exploiter au mieux la transmission sur la voie haute vitesse.

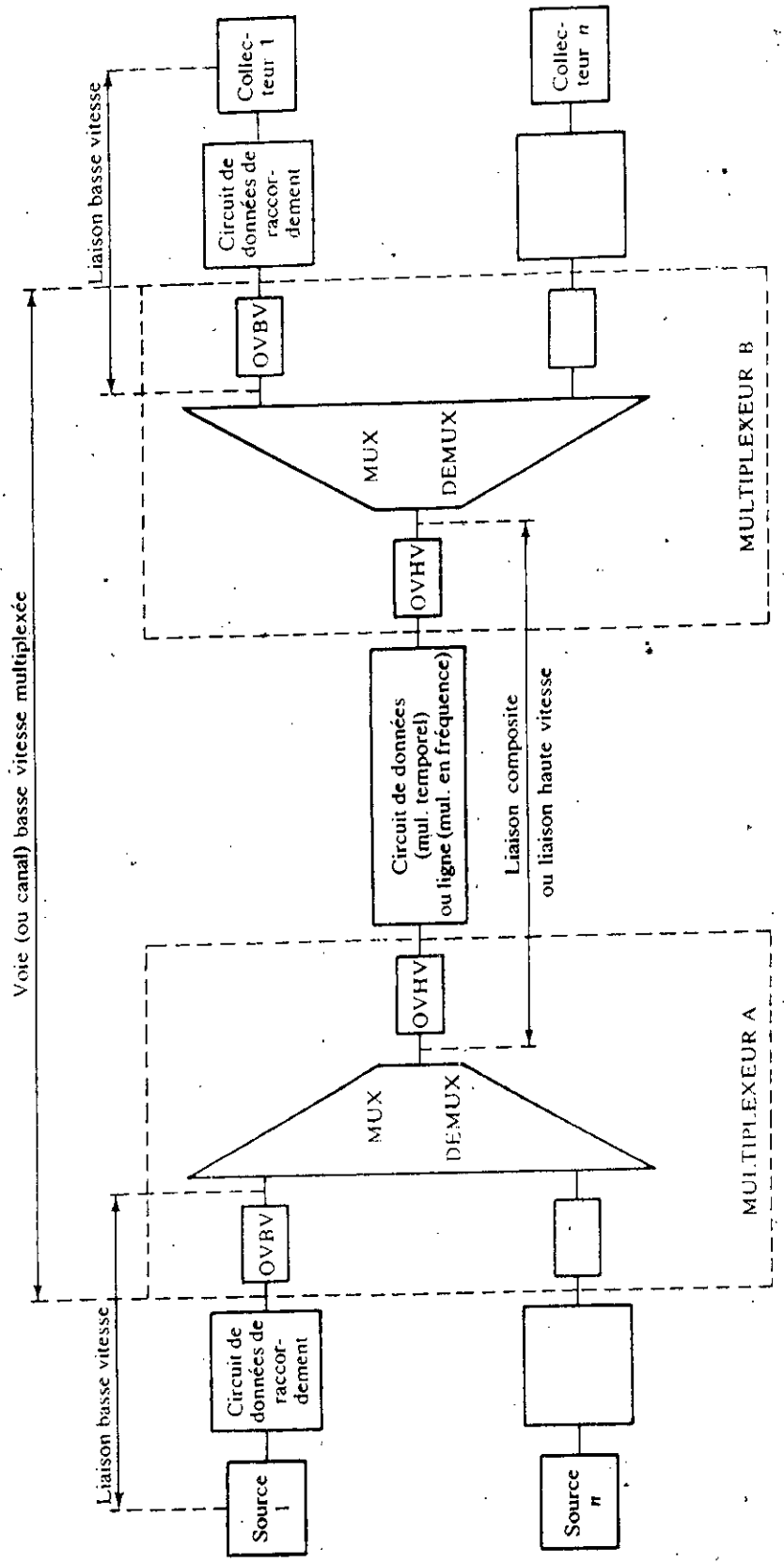


Figure 1.1. Constitution d'une liaison utilisant des multiplexeurs

— L'aptitude à mélanger des voies de caractéristiques différentes (différentes par le code, par la vitesse, par le mode de transmission - synchrone ou asynchrone - ...).
— Le transfert de voies: les données d'une voie du site A vers le site C peuvent être, soit démultiplexées et remultiplexées dans le site B soit transférées directement; on dit alors que B effectue le transfert de voies BV entre deux liaisons composites. Ceci permet de réduire le coût en matériel.
— La transmission des signalisations: les signalisations sont les informations de test et de maintenance pour le dialogue entre les multiplexeurs ou entre les équipements connectés aux deux extrémités du canal. Deux solutions sont possibles:
  — hors bande: sur un canal multiplexé séparé, réservé à cet effet, appelé canal sémaphore,
  — dans la bande: sur le canal lui-même, à la place des données.

2.1.- MULTIPLEXAGE STATIQUE

Ce sont des équipements non programmables, le plus souvent câblés. Ils effectuent un multiplexage en fréquence ou en temps et sont transparents aux flots de données qui les traversent.

2.1.1.- Multiplexage en fréquence

Le principe général du multiplexage en fréquence dans le cas d'une transmission de données, consiste en la transformation des suites de données exprimées par les signaux di(t) bivalents de chaque voie basse vitesse i en signaux sinusoïdaux selon une transformation du type:

$$di\ (t) \equiv \begin{cases} 0 \text{ ---> } \sin 2\pi\ (fi + W1)t \\ 1 \text{ ---> } \sin 2\pi\ (fi - W1)t \end{cases}$$

Les paires de fréquences (fi + W1) correspondant aux différentes voies basse vitesse, sont choisies de telle sorte qu'elles se répartissent sans se chevaucher dans la bande passante d'une voie téléphonique normale (300-3400 Hz). A la réception, une batterie de filtres passe-bande sépare les voies puis des discriminateurs de fréquence restituent les signaux bivalents.

Le CCITT recommande les valeurs suivantes, où i désigne le rang de la voie considérée :

A 50 bauds : fi = 420 + (i - 1) 120 Hz et W1 = 30 Hz
ce qui permet de réaliser 24 voies.

A 100 bauds : fi = 480 + (i - 1) 240 Hz et W1 = 60 Hz
ce qui permet de réaliser 12 voies.

A 200 bauds : fi = 600 + (i - 1) 480 Hz et W1 = 120 Hz
ce qui permet de réaliser 6 voies.

Les multiplexeurs en fréquence transmettent en général une signalisation dans chaque sens (Half-duplex). Ils peuvent mélanger des voies de vitesses différentes. Ils sont limités dans leurs possibilités aux vitesses de 50, 100 et 200 bauds. Leur efficacité est faible, voisine de 0.20. Le transfert de voie est possible en gardant la même fréquence porteuse sur les deux liaisons composites.

2.1.2.- Multiplexage en temps

Le multiplexage temporel consiste à allouer à chaque voie basse vitesse toute la ressource de la voie composite, pendant une partie du temps, multiple d'un intervalle de temps élémentaire, au prorata du débit de chaque terminal.

La somme des vitesses nominales des voies basse vitesse ne doit jamais excéder la vitesse de la voie composite.

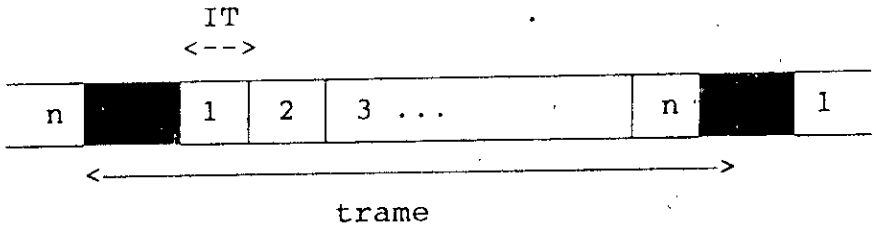
2.1.2.1.- Multiplexage temporel par caractère

Cette technique est particulièrement adaptée à des voies basse vitesse asynchrones; elle est non transparente au code et à la vitesse.

Le train de bits de débit D sur la voie haute vitesse est divisé en trames de L intervalles de temps (IT).

Chaque IT est réservé à un caractère en provenance d'une voie basse vitesse (débarrassé des bits START et STOP dans le cas

asynchrone) plus un bit indiquant la nature du caractère transmis



(information ou signalisation). La longueur de la trame peut être tirée à partir de la formule suivante:

$$L\ (IT) \leq \frac{D\ (car/s)}{Di\ (car/s)}$$

Lorsque les Di sont égaux pour les n voies basse vitesse, L est égal à n.

Pour pouvoir reconnaître les IT, il faut reconnaître le début d'une trame: le premier IT est réservé, il contient une combinaison particulière. En cours de transmission, le récepteur teste en permanence la présence de ce caractère, appelé caractère de verrouillage de trame. Si le test est négatif deux fois de suite, le récepteur bloque l'émission des données vers les voies basse vitesse et génère un signal d'alarme (perte de synchronisation).

Pour la transmission de signalisation, il suffit de choisir Ai = li + 1. Le bit supplémentaire indique, selon sa polarité, si les li autres bits contenus dans l'IT sont des données ou des informations de signalisation: changement d'état sur la jonction basse vitesse, commandes de bouclage, informations concernant l'état de transmission de la voie basse vitesse.

Lorsque les voies basse vitesse sont synchrones, il est obligatoire que la ligne haute vitesse et les lignes basse vitesse soient synchronisées rigoureusement entre elles, c'est à dire que leurs horloges soient toutes des multiples entiers d'une horloge de base. Il doit donc y avoir une horloge unique pilotant l'ensemble du réseau.

Le mélange de voies basse vitesse de débits différents sur un

même circuit composite est possible. Une première solution consiste à organiser la trame en un nombre d'IT correspondant au plus grand des débits basse vitesse et à les utiliser pour faire passer les voies de débit inférieur. Cette solution est simple mais diminue l'efficacité du multiplexage. Une autre méthode consiste à culculer la longueur L de la trame, en fonction du débit de la voie basse vitesse le plus faible et à affecter au même canal plusieurs IT de la même trame, régulièrement espacés.

Le transfert de voie est généralement possible pour tous les multiplexeurs temporels.

Les multiplexeurs temporels par caractère ont une efficacité bien meilleure que les multiplexeurs en fréquence; elle est de l'ordre de 0.8. Ils acceptent les débits basse vitesse allant, pour certains d'entre eux, de 50 à 19200 bit/s, en mode asynchrone et de 1200 à 56000 bit/s en mode synchrone. Le débit de la voie haute vitesse peut atteindre 72 kbit/s pour certains équipements.

2.1.2.2.- Multiplexage temporel par bit

Le multiplexage temporel par bit est une technique plus particulièrement adaptée au cas des basses vitesses synchrones.

Le principe est identique à celui du multiplexage par caractère. La seule différence est que les IT ont une longueur de 1 bit. La longueur de la trame est calculée d'après le débit des voies basse vitesse, exprimé en bits/seconde et non plus en caractères/seconde.

De même que précédemment le premier IT est réservé, il constitue un canal basse vitesse non affecté mais transportant en permanence une séquence de plusieurs bits connus.

La signalisation est codée pour chaque voie basse vitesse sur 8 bits (5 pour les informations de jonction et 3 pour les commandes de bouclage). Suivant le constructeur, elle est faite hors bande ou dans la bande:
— hors bande: dans la trame on réserve un IT supplémentaire qui permet de transmettre les signalisations successives de chaque voie basse vitesse. Cette méthode est complexe pour le transfert de voies.

— dans la bande: la signalisation est émise sur le canal lui-même précédée d'une séquence spéciale préparatoire. Cette méthode est idéale pour le transfert des voies mais restreint la transparence: le multiplexeur recherche en permanence la séquence spéciale et crée une erreur systématique si elle apparaît dans les données.

Le mélange de voies de vitesses différentes est réalisé par l'affectation de plusieurs IT à un même canal. Il est alors nécessaire que les débits binaires des voies basse vitesse soient tous des multiples entiers et exacts du débit du canal le moins rapide.

Les multiplexeurs temporels par bit ont une efficacité sensiblement identique à celle des multiplexeurs par caractère. A titre d'exemple, il est possible de multiplexer sur un circuit exploité à 72 kbit/s, 58 voies à 1200 bit/s avec une efficacité de 0.97.

2.2.- MULTIPLEXAGE TEMPOREL STATISTIQUE

Comparés aux multiplexeurs en fréquence, les multiplexeurs temporels se caractérisent par une efficacité bien meilleure. Cependant, lorsqu'on sait que sur la majorité des liaisons de transmission de données, les durées de transmission effectives ne représentent que 10 à 30% du temps global de connexion, on est obligé de reconnaître que même les multiplexeurs temporels sont loin d'optimiser à 100% les possibilités de transmission sur la liaison composite. Une nouvelle génération s'apparentant plus aux concentrateurs qu'aux multiplexeurs, est alors apparue, permettant d'améliorer la concentration sur la liaison composite. Ce sont les multiplexeurs statistiques.

Leur originalité par rapport aux multiplexeurs classiques consiste à allouer dynamiquement les intervalles de temps d'une trame aux voies basse vitesse qui sont actives à un instant donné. Ceci leur permet d'utiliser, les intervalles de silence (représentant 70% du temps de connexion, selon des statistiques) qui existent sur toute liaison de transmission de données. Le gain d'efficacité important qui en résulte est mis à profit pour accueillir le trafic d'un nombre accru de voies et pour gérer une procédure de transmission par paquets avec détection et récupération des erreurs sur la liaison composite.

2.2.1.- Fonctionnement

Les multiplexeurs statistiques sont organisés autour d'un ou plusieurs microprocesseurs gérant des organes de voies connectés aux voies basse vitesse, et des organes de voies connectés aux liaisons composites. Ils disposent de mémoires vives pour assurer un stockage temporaire des données. La taille de cette mémoire varie selon le matériel et le nombre de voies basse vitesse gérées; elle est typiquement de 16K à 100K octets.

Le message binaire provenant d'une voie basse vitesse est assemblé en caractères, conformément aux caractéristiques de code et vitesse de la ligne. Chaque caractère est débarrassé de tous ses éléments «inutiles» (START, STOP, bit de parité) qui seront régénérés par le multiplexeur distant.

Par ailleurs, les changements des signaux de contrôle sur la jonction entre l'ETTD (Equipement Terminal de Traitement de Données) et l'OVBV (Organe de Voie Basse Vitesse), sont aussi codés sous forme de caractères. Caractères de données et caractères de signalisation sont stockés dans une mémoire tampon allouée à la voie basse vitesse par le microprocesseur dans une zone de mémoire commune. Cette allocation a lieu lorsque la voie devient active.

La longueur des tampons est fixée pour chaque canal, par exemple, en fonction croissante du débit binaire. Un système de pointeurs permet au besoin de chaîner plusieurs tampons pour une même voie.

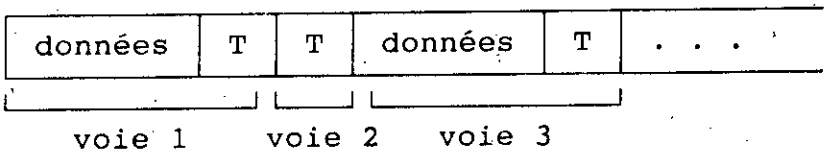
La reconnaissance de la mise en activité ou de la fin d'activité d'une voie est un problème majeur du multiplexage statistique. Elle dépend de la nature des terminaux ou de leur mode de connexion:
— établissement de la connexion au multiplexeur, lorsque l'accès à celui-ci a lieu par réseau commuté,
— reconnaissance de caractères spéciaux (synchronisation, début ou fin de message) lorsque la source est un terminal travaillant en mode message. Ceci suppose que le multiplexeur sache interpréter les procédures de communication employées par chaque terminal susceptible de l'appeler.

Lorsque la liaison composite est prête à transmettre une nouvelle trame, le processeur explore les mémoires tampons des

voies actives et transfère leur contenu dans une mémoire spécifiquement allouée à l'OVHV (Organe de Voie Haute Vitesse). Il constitue ainsi un message (la trame) qui sera transmis au multiplexeur distant. La constitution de la trame peut être faite de différentes manières:

— une première méthode, simple, consiste à réserver les deux premiers octets de la trame, l'un au numéro de la voie basse vitesse concernée et l'autre, au nombre de caractères transférés pour cette voie. Aucune place n'est allouée aux voies inactives;

— une autre méthode consiste à prendre des trames de longueur fixe et à attribuer à toutes les voies basse vitesse actives ou non, un ordre déterminé sur la voie composite. Un séparateur de deux bits sépare les trames de deux voies successives. Si une voie est inactive, aucune donnée n'est placée pour cette voie sauf le séparateur. Cette méthode est rendue possible grâce au code de Huffman.



T= séparateur de 2 voies (2 bits dans le code de Huffman).

Ainsi constituée, la trame est émise vers le multiplexeur distant sur la liaison composite, exploitée en duplex selon une procédure de communication, le plus souvent de type HDLC X25 niveau 2. Une séquence de contrôle de trame permet de détecter les erreurs de transmission et de demander la répétition des trames reconnues erronées à la réception.

En plus des fonctions de multiplexage proprement dites, le multiplexeur statistique calcule et édite un certain nombre de renseignements sur l'état de fonctionnement du réseau: statistiques d'erreurs de transmission, de charge des lignes, taux d'utilisation des mémoires tampon, etc...

Enfin, une console opérateur permet de modifier les paramètres de transmission, en particulier de commander un changement de programmation (acheminement des différents canaux au sein d'un réseau), tant sur le multiplexeur local que sur les multiplexeurs distants.

2.2.2.- Performances

Les multiplexeurs statidtiques sont tout d'abord caractérisés par une efficacité allant de 2 à 4, bien meilleure que celle des multiplexeurs classiques. A titre d'exemple, on peut connecter plus de 100 voies à 110 bauds sur une seule liaison composite exploitée à 4800 bit/s. Par contre, il y a un risque de saturation (donc de perte d'informations) si tous les terminaux sont actifs simultanément. La probabilité en est très faible, comme le montrent, en particulier, les travaux de Wesley W. Chu. Elle est négligeable dans la majorité des cas classiques d'exploitation, et en tous cas peut être estimée à l'avance en fonction du nombre de terminaux raccordés et de leur trafic moyen. Si un tel cas de saturation se produit, le multiplexeur peut éventuellement contrôler le flux des données émises par les sources en envoyant vers celles-ci des ordres d'arrêt (X-OFF), puis de reprise de la transmission (X-ON).

Les multiplexeurs statistiques transmettent les signaux de contrôle des jonctions. Ils sont transparents aux codes et aux procédures. Ils offrent des possibilités de gestion et de contrôle du réseau généralement supérieures à celles des multiplexeurs temporels classiques. Ils permettent la réalisation de transferts de voies, ainsi que de canaux multipoints. Ils offrent une très bonne résistance aux erreurs de transmission sur les liaisons composites (grâce à la gestion d'une procédure sur celles-ci).

Par contre, du fait de la mémorisation des données dans les mémoires de voies, avant émission de la trame, ils introduisent un retard dans la transmission qui peut être beaucoup plus important que les multiplexeurs traditionnels. Comparés aux concentrateurs, ils ont l'avantage d'être totalement transparents, et donc ne nécessitent aucune adaptation de logiciel, ni dans les terminaux, ni dans les ordinateurs.

2.3.- POSSIBILITES DES DIFFERENTS TYPES DE MULTIPLEXEURS [01]

Nous présenterons dans le tableau 1.1 les possibilités des différents types de multiplexeurs traités ci-dessus dans les domaines de:

— synchronisation d'un réseau, noté S.R.,
— gestion de réseau, noté G.R.,
— sensibilité aux erreurs de transmission, noté S.T.,

— retard de transmission, noté R.T.

Tableau 1.1.

	Multiplexeurs en fréquence	Multiplexeurs temporels	Multiplexeurs statistiques
S.R.	- Pas de problème de synchronisation.	- Pas de problème de synchronisation quand les voies à concentrer sont asynchrones. - Quand les voies à multiplexer sont synchrones l'horloge de la voie composite et celles des voies basse vitesse doivent être pilotées par une horloge unique.	- La synchronisation n'est pas toujours obligatoire. La présence des mémoires tampons peut compenser dans certains cas des différences importantes d'horloge, à condition que la transmission se fasse par messages de longueur finie et non en permanence.
G.R.	- Capacités d'administration de réseau réduites à la détection et la signalisation de la coupure du circuit composite. - Toute modification des caractéristiques d'une voie basse vitesse nécessite une intervention humaine à chaque extrémité pour changer l'OVBV correspondant.	- Ils disposent de certaines facilités de maintenance: détection des coupures de la liaison composite, possibilité d'injecter des séquences de test, de contrôler le bon fonctionnement du multiplexeur local, etc... - Possibilité de modifier les paramètres de configuration du multiplexeur local et télécharger les modifications correspondantes dans les multiplexeurs distants qui lui sont reliés.	- Tout le réseau est géré à partir d'un point central: les opérations de configuration, de contrôle et de test sont commandées à distance.Les déplacements et interventions humaines se limitent donc, au remplacement des matériels défectueux. - Des statistiques de fonctionnement et de charge sont tenues par chaque multiplexeur et peuvent être centralisées en un point du réseau sur un journal de bord. - Possibilité d'être connecté à n'importe quelle voie par simple message. - Possibilité d'être lié à plusieurs voies en même temps. - Etc...
S.T.	- Peu sensibles aux bruits impulsifs et aux micro-coupures qui se produisent sur le circuit de données haute vitesse. - Peu sensibles également, aux déplacements de fréquence dus à une mauvaise stabilité des porteuses.	- Insensibles aux déplacements de fréquence, mais par contre très sensibles aux bruits et micro-coupures. Les multiplexeurs par bit le sont encore plus que les multiplexeurs par caractère: un bruit affectant 8 bits de la trame, perturbe au plus 2 canaux d'un multiplexeur par caractère, mais perturbe 8 canaux d'un multiplexeur par bit.	- Pratiquement insensibles à toutes les perturbations de faible durée: ils gèrent une procédure sur la voie composite qui leur permet de récupérer les erreurs de transmission.
R.T.	- Ils n'introduisent aucun retard dans la transmission. - Ils supportent certaines variations autour de la vitesse nominale du canal, dans la limite de tolérance des filtres de réception.	- Ils introduisent un retard systématique dans la transmission, car l'information doit être stockée en attendant d'être placée dans un IT de la trame. Le retard introduit par les multiplexeurs par bit est toujours inférieur à celui introduit par les multiplexeurs par caractère.	- Ils introduisent un retard d'autant plus important que la taille des mémoires tampons est grande.

3.- LES CONCENTRATEURS [01, 02]

Tout comme le multiplexage, la concentration a pour rôle essentiel de regrouper sur un circuit composite unique des informations provenant de plusieurs circuits de données.

Le concentrateur est un microcalculateur programmé, qui gère une procédure de communication avec les terminaux sources et stocke temporairement les données utiles. La somme apparente des débits entrants peut être supérieure à celle des débits sortants puisque le concentrateur permet de supprimer les silences qui représentent une part souvent très importante de la durée des communications dans les applications de téléinformatique.

Parmi les fonctions qu'il assure, nous retrouvons:
— la détection et la correction des erreurs sur la ligne haute vitesse et éventuellement sur les lignes basse vitesse,
— la désynchronisation complète entre le trafic de la ligne haute vitesse et celui des lignes basse vitesse, ce qui lui permet de réaliser entre autres, toutes les conversions de code ou de vitesse nécessaires, l'acheminement des informations sur la ligne haute vitesse la moins chargée, etc,
— la centralisation des informations sur l'état des lignes basse vitesse et des équipements qui leurs sont raccordés et leur transmission au calculateur central,
— la suppression ou génération des informations répétitives ne possédant pas de valeur intrinsèque, comme les en-têtes de document, date, grilles de positionnement, etc..., ceci permet de ne transmettre sur la ligne haute vitesse que l'information utile,
— gestion de l'activité des différents terminaux et éventuellement guidage pas à pas des opérateurs chargés de la saisie sur ceux-ci.

Les principales caractéristiques d'un concentrateur sont les suivantes :
— miniordinateurs dont l'architecture est orientée vers les transmissions avec un ensemble d'instructions réduit, mais un cycle d'exécution très rapide,
— capacité de mémoire vive peu importante,
— dispositifs de stockage de masse peu développés ou inexistants,
— système d'exploitation n'assurant que des fonctions simples, mais permettant de gagner le maximum de rapidité d'exécution,
— dispositifs d'interruption et d'entrée-sortie de transmission très développés.

4.- PROTOCOLE DE COMMUNICATION SDLC [01, 02]

La communication sur la voie haute vitesse est assurée par le protocole SDLC (Synchronous Data Limit Control). Cette procédure de liaison introduite par IBM pour son architecture de réseau SNA est plus performante que ses prédécesseurs «en mode de base». La structure de la trame est représentée comme suit:

DRAPEAU	ADRESSE	COMMANDE	DONNEES	FCS	DRAPEAU

Figure 1.2. Structure de la trame SDLC

Le drapeau indique le début ou la fin de la trame. Il est toujours formé de la suite 01111110. Pour que la même suite ne se retrouve pas dans la trame, on insère automatiquement un 0 dès que l'on trouve une suite de cinq 1. Le récepteur agit de la même façon: s'il trouve un 0 après cinq 1, il l'élimine; s'il trouve un 1 c'est un drapeau.

Le champ d'adresse contient l'adresse de la station secondaire; elle tient sur un octet.

Le champ de commande permet à la fois le numérotage de la trame envoyée et l'envoi des accusés de réception. Pour cela, il est structuré en trois formats de transmission.

Bit	1	2	3	4	5	6	7	8
Format d'information (I)	I	0	N(S)			P/F	N(R)	
Format de supervision (S) commande RR (Receive Ready)	S	1	0	0	0	P/F	N(R)	
commande RNR (Receive Not Ready)				1	0			
commande REJ (Reject)				0	1			
Format non numéroté (U)	U	1	1	M		P/F	M	

N(S) = numéro de la trame I émise.

N(R) = numéro de la trame attendue par le récepteur.
P/F = bit d'invitation à émettre.
M = bits de réserve pour commandes supplémentaires

La zone de contrôle FCS détecte les erreurs de transmission sur la trame. Celle-ci est par conséquent détruite et un trou dans la numérotation N(R) se produit. Le récepteur réclame alors la trame qui n'a pas été reçue ainsi que toutes les trames suivantes.

Les caractéristiques essentielles du protocole SDLC sont les suivantes:
— ils permettent une exploitation duplex de la liaison de données,
— les trames peuvent contenir à la fois des données et des informations de service, telles que les accusés de réception,
— plusieurs trames peuvent être émises en séquence, sans qu'un accusé de réception soit envoyé pour chacune d'elles,
— toutes les trames, même sans données, sont protégées des erreurs de transmission par un FCS (Frame Check Sequence),
— le protocole SDLC assure la transmission de suites d'éléments binaires et non de caractères.

5.- Principales caractéristiques techniques des multiplexeurs statistiques

Les principales caractéristiques techniques des multiplexeurs temporels statistiques sont les suivantes:

— Multiplexage statistique de canaux asynchrones.
— Canaux basse vitesse jusqu'à 9600 bauds.
— Canaux haute vitesse jusqu'à 19200 bauds.
— Retransmission automatique en cas d'erreur.
— Détection automatique de code et de vitesse en standard.
— Transmission de signaux de contrôle pour chaque canal.
— Régulation du flot de données en standard.
— Protection par mémoire tampon contre les erreurs ou les coupures de la ligne haute vitesse.
— Téléchargement et configuration par logiciel.
— Différentes priorités des voies basse vitesse.
— Passage aisé d'un modèle au suivant et compatibilité ascendante.
— Validation de chaque canal sans interruption de données.
— Surveillance automatique et continue du système.
— Gamme complète de possibilités de bouclage et de diagnostic.

CHAPITRE 2

DEVELOPPEMENTS THEORIQUES

Dans le but de réduire le coût des communications dans les systèmes de communication en temps partagé et multiutilisateurs à distance, des techniques de multiplexage furent introduites pour optimiser l'utilisation des canaux de transmission de données. La technique la plus performante et la plus efficace, utilisée de nos jours, est le multiplexage temporel statistique. Ses deux principales fonctions sont le multiplexage et le démultiplexage. Le multiplexage consiste en une opération d'acquisition de données émanant de sources multiples, leur organisation, puis leur orientation vers une destination unique. Le démultiplexage, quant à lui, se charge de l'acquisition des données arrivant d'une source unique, pour les réorganiser puis les distribuer aux destinataires appropriés. Les paramètres cruciaux de ce type de multiplexage sont:

— une adresse requise pour chaque message émis,
— une bufferisation requise pour la sauvegarde des messages qui arrivent.

La mise en oeuvre de ce multiplexage dépend:

— d'une probabilité de débordement réduite, compensée (ou éliminée) par une taille mémoire tampon (bufferisation) raisonnable,
— d'un temps d'attente des données dans la mémoire tampon, acceptable.

De nombreuses études ont été réalisées pour l'estimation de ces paramètres et des recherches sont menées sur ce sujet qui continue à susciter l'intérêt de beaucoup de chercheurs, notamment sur les applications étendues aux réseaux de communication. Nous tâcherons de présenter dans ce qui suit, les principaux travaux effectués dans ce domaine, entre les années 1970 et 1990.

1.- MODELE DE MULTIPLEXAGE [65, 44]

Le multiplexage statistique procède par test circulaire sur tous les utilisateurs, y compris ceux qui sont inactifs, puis multiplexe la donnée, de manière temporelle et asynchrone, de telle sorte que chaque usager ne dispose d'une garantie d'accès au canal composite que lorsqu'il possède une information à émettre.

Dans ce modèle, élaboré dans les années 1970 par W.W. Chu [65, 44], les arrivées «caractères» débitées des sources utilisateurs dans le tampon de réception (nous considérons un même tampon pour la réception des caractères des différents utilisateurs) sont caractérisées par des temps inter-arrivées indépendants et identiquement distribués. Pour l'application numérique, ces arrivées seront supposées poisoniennes de distribution:

$$f_n = \exp(-\lambda)\lambda^n / n!$$

où λ représente le taux des arrivées des caractères dans le buffer de réception à partir des différentes sources indépendantes.

La sortie des caractères de ce buffer se fera de manière synchrone: ces derniers seront transmis à des tops d'horloge fixes. Ceux qui arrivent entre temps sont stockés dans la mémoire tampon, dans l'attente du prochain top d'horloge, même si le serveur est libre. Dans la terminologie de la théorie des files d'attente, on dit que le service est régulé par une porte située entre le serveur et la file d'attente et ouverte à des intervalles fixes. Comme la ligne synchrone pédale à une vitesse constante, le temps écoulé dans le transfert d'un caractère (temps de service) sera supposé constant et unitaire.

Comme la taille de la mémoire tampon est finie, de longueur N, si un caractère arrive et trouve cette mémoire pleine, un débordement se produit. La probabilité de ce débordement P_{of} , est égale à:

$$P_{of} = 1 - \alpha/\lambda$$

avec:

$$\alpha = 1 - p_0$$

où p_0 est la probabilité que le tampon est vide.

Ainsi, la longueur moyenne de la file d'attente, est égale à:

$$L = \sum_{i=1}^{N} (i-1)p_i + \lambda/2 \quad \text{caractères}$$

et le temps d'attente moyen d'un caractère dans la file d'attente, en est déduit grâce à la formule de Little, comme suit:

$$W = \frac{L}{\lambda(1-P_{of})} \quad \text{unités de temps}$$

La figure 2.1 traduit l'évolution de la probabilité de débordement en fonction de la taille du tampon, pour des intensités de trafic données. Pour une intensité de trafic donnée, la probabilité de débordement décroît de manière exponentielle avec la taille du tampon.

Les figures 2.2 et 2.3 montrent l'allure du temps d'attente moyen en fonction de la taille du tampon et de l'intensité du trafic respectivement.

Calculons à titre d'exemple, ce que ceci pourrait donner pour un système (le nôtre), dont le nombre d'usagers sur un site est de huit, fonctionnant chacun à une vitesse de 1200 bits/seconde soit 120 caractères à la seconde. La vitesse de la voie composite est de 9600 bits/seconde soit 1200 caractères à la seconde. Ainsi, l'intensité du trafic serait:

$$\rho = m\lambda/\mu = 8 \times 120 / 1200 = 0.8$$

Avec une intensité du trafic égale à 0.8 et une probabilité de débordement de 10^{-10} , les résultats de la figure 2.1 donnent une taille du tampon égale à une cinquantaine de caractères. Si l'on réduisait davantage cette probabilité de débordement jusqu'à atteindre l'ordre de 10^{-17} , la taille du tampon atteindrait les quatre vingt dix caractères.

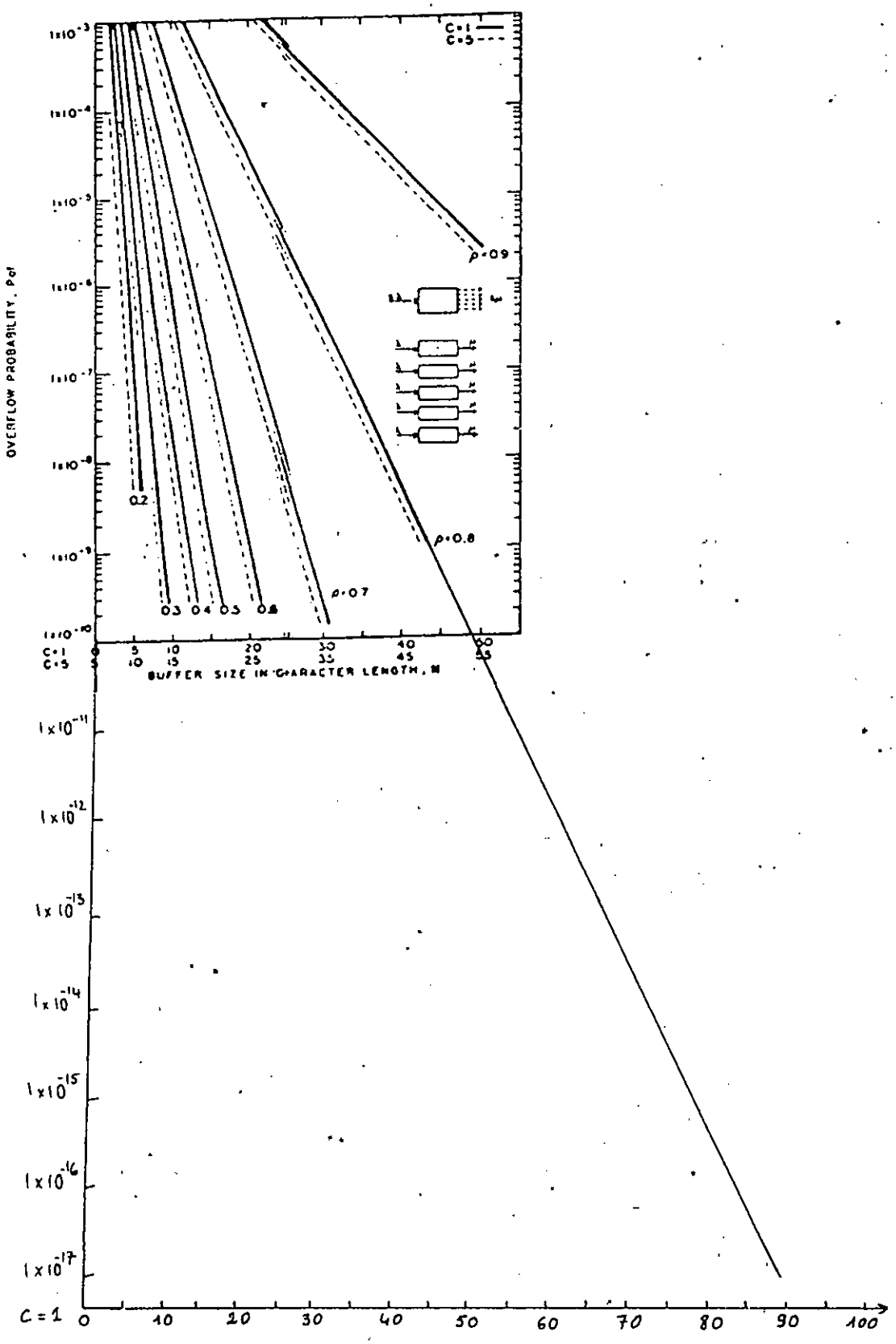


Figure 2.1. Probabilité de débordement en fonction de la taille du tampon.

Quant à la figure 2.3, elle fournit pour une intensité de trafic de 0.8, un temps d'attente moyen de 2.5 unités de service. Comme une unité de service est égale à $1/\mu = 1/1200 = 833.33\ \mu s$, le temps d'attente moyen de chaque caractère est de 2.1 millisecondes.

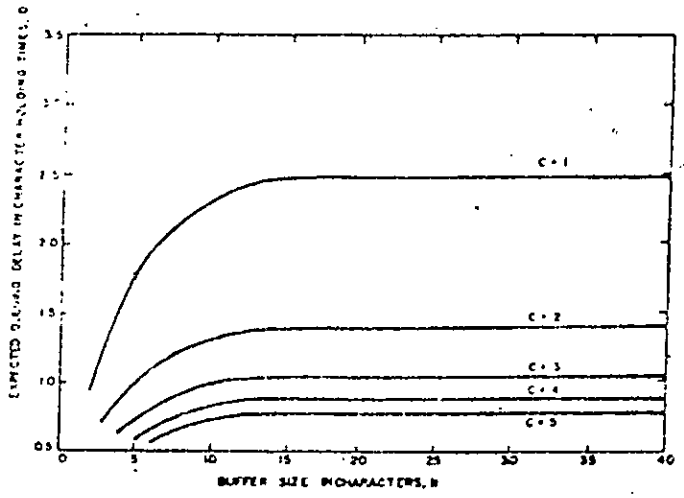


Figure 2.3. Temps d'attente moyen dans la file en fonction de la taille du tampon.

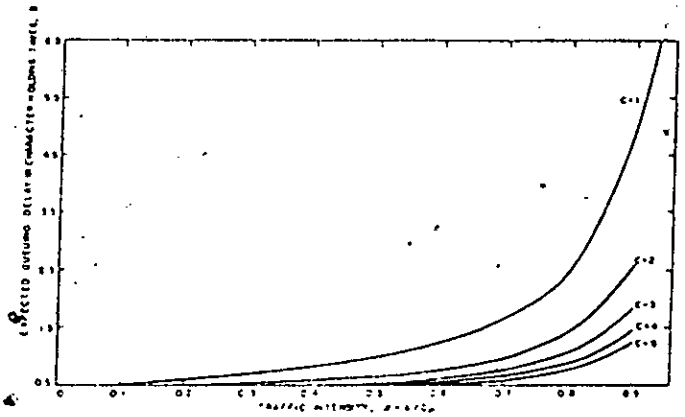


Figure 2.3. Temps d'attente moyen dans la file en fonction de l'intensité du trafic.

2.- MODELE DE DEMULTIPLEXAGE [46, 49]

Le tampon de démultiplexage ou mémoire tampon d'émission a été modélisé en 1972 par W.W. Chu [49], comme un processus de file d'attente de taille finie, avec des arrivées poissoniennes et de multiples et distinctes sorties constantes. Les messages d'entrée dans le tampon sont des chaînes de caractères (ou trames). La complexité de l'étude du comportement de ce tampon rend extrêmement difficile l'application de l'analyse mathématique exacte. C'est pourquoi, une étude par simulation est effectuée pour déterminer la relation entre la fonction de destination des messages, la moyenne du niveau de trafic, la longueur des trames, la probabilité de débordement et la taille du tampon.

Les trames arrivant dans le buffer sont représentées par trois paramètres:

— la fréquence d'arrivée des messages,
— la longueur des messages, et
— la destination des messages.

Ces trois paramètres influencent considérablement le comportement de la mémoire tampon. Ils seront décrits par les trois variables aléatoires suivantes: α pour la durée des interarrivées, β pour le nombre de caractères de la trame et τ pour la destination de la trame. Dans le modèle de simulation, les durées des inter-arrivées α, seront supposées exponentiellement distribuées, les longueurs des messages β, seront supposées géométriquement distribuées. Quant à la distribution de la destination τ, elle sera tirée de la fonction de destination.

La fonction de destination décrit l'intensité du trafic pour les m destinations:

$$f_d(i) = \rho_i \qquad i=1,2,\ldots,m \qquad (1).$$

où ρ_i est l'intensité du trafic de la ième destination. La transformation de la fonction de destination $f_d(i)$ en une distribution de la destination du message, $f_\tau(i)$, peut se faire par une normalisation de (1), telle que:

$$f_r(i) = f_d(i) \Big/ \sum_{i=1}^{m} \rho_i \qquad i=1,2,\ldots,m$$

donc:

$$\sum_{i=1}^{m} f_r(i) = 1$$

Lorsque la taille des messages qui arrivent excède celle de la mémoire tampon, un débordement de ce tampon se produit. La fréquence de ces débordements conduit à l'estimation de la probabilité de débordement du tampon d'émission P_{of} . L'estimation de cette probabilité de débordement à l'aide d'un modèle dont la taille du tampon est finie, requiert la simulation du comportement du tampon pour différentes tailles mémoire, ce qui mène à des coûts de simulation très élevés. C'est pourquoi, un modèle dont la taille du tampon est infinie, est utilisé. Ce modèle fournit une bonne approximation du modèle tampon-de-taille-finie, lorsque les probabilités de débordement sont petites ($P_{of} < 10^{-4}$).

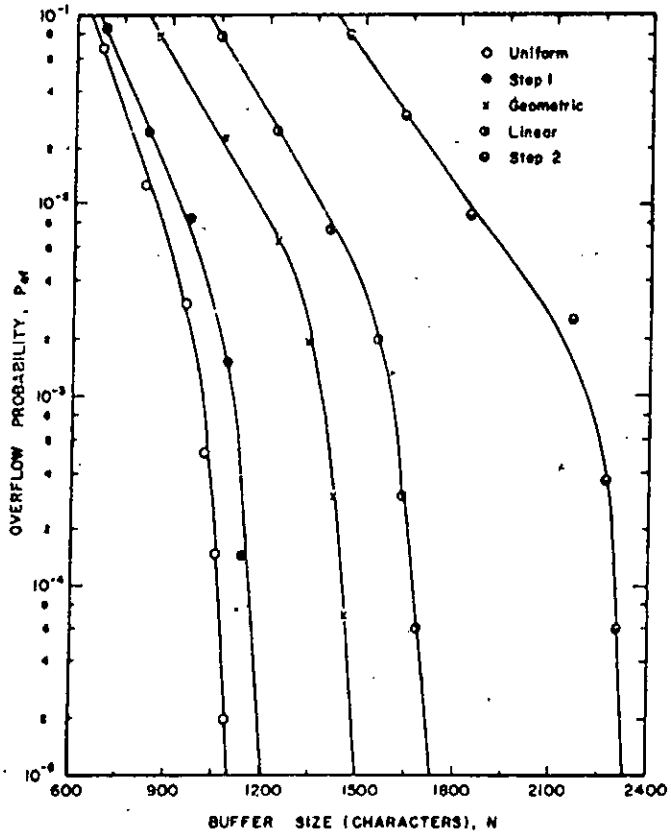
L'intensité du trafic est l'un des plus importants paramètres décrivant la congestion du système. Comme les messages sont orientés vers des destinations différentes, les intensités de trafic sur ces destinations seront:

$$\rho_i = \lambda_i l_i / \mu_i \qquad \text{et} \qquad \bar{\rho} = 1/m \sum_{i=1}^{m} \rho_i$$

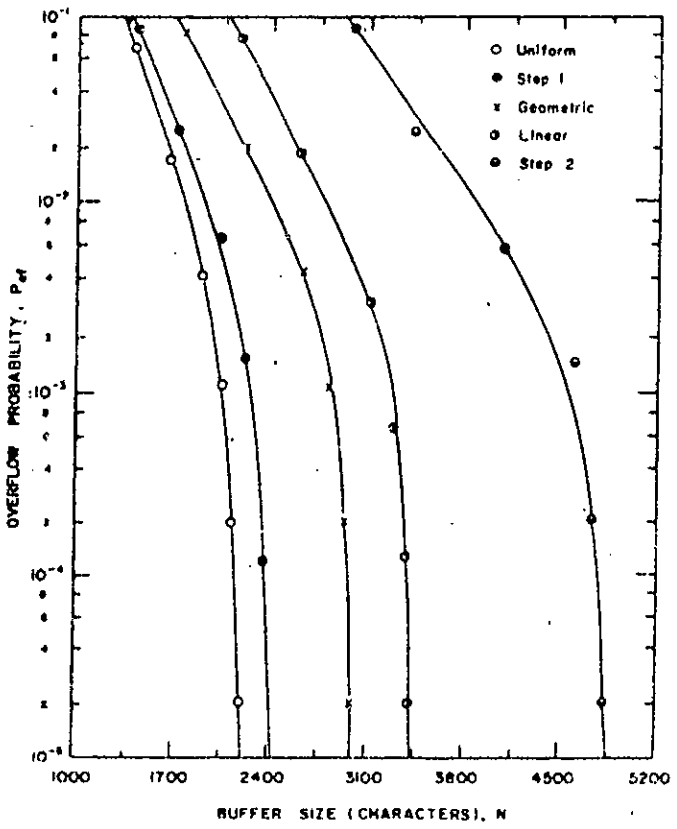
où:

λ_i = taux des arrivées sur la $i^{ème}$ destination
l_i = longueur moyenne de la trame de la $i^{ème}$ destination
μ_i = taux d'émission sur la $i^{ème}$ destination

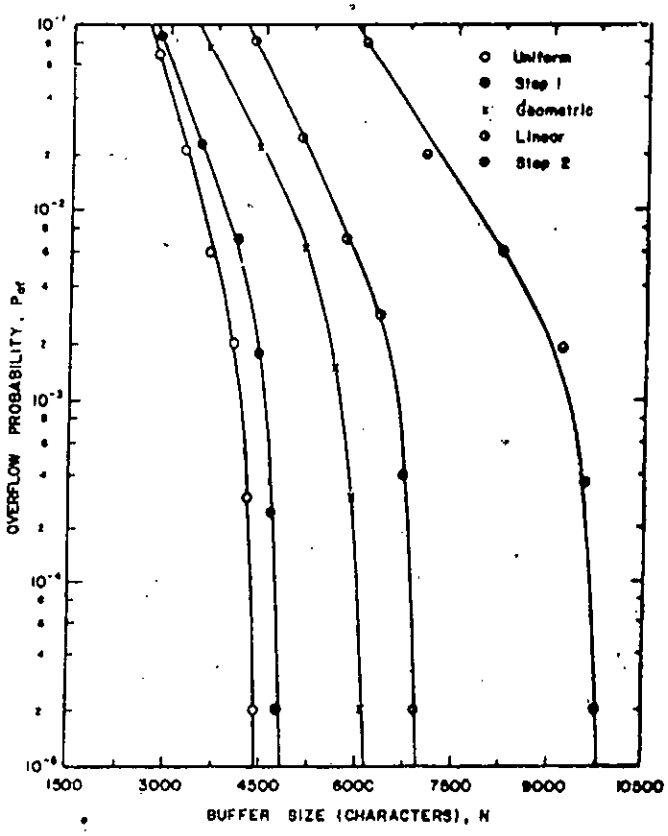
Pour étudier l'effet des distributions de la destination sur le comportement du tampon, cinq types de fonctions de destination sont utilisées dans le modèle de simulation, à savoir: les fonctions uniforme, linéaire, pas de 1, pas de 2 et géométrique. De plus, pour écarter l'effet de la variabilité de la longueur de la trame selon la destination, sur le comportement du tampon, on suppose la longueur moyenne de la trame, égale pour les différentes destinations, $l_i = l = 1/\Theta$, pour i= 1 à m. Les relations entre P_{of} , la taille du tampon et les distributions de la destination pour des valeurs de l et ρ données, ont été déterminées par simulation et sont représentées sur les graphes de la figure 2.4.



(a)



(b)



(c)

Figure 2.4. Probabilité de débordement en fonction de la taille du tampon, pour une moyenne de l'intensité du trafic de σ=0.8. (a) l=10 caractères. (b) l=20 caractères. (c) l=40 caractères.

Ainsi, pour une intensité de trafic de 0.8, une moyenne de la longueur de la trame de 20 caractères (figure 2.4 b), une probabilité de débordement de 10^{-5} et une distribution uniforme de la longueur de la trame, la taille du tampon sera de 2200 caractères.

Le temps d'attente du à la bufferisation durant une opération de démultiplexage, est un autre paramètre important pour les multiplexeurs statistiques. Comme la majorité des systèmes permet une très petite probabilité de débordement, le temps d'attente du à la bufferisation peut être approximé par celui correspondant à un modèle dont la taille du tampon est infinie avec des arrivées poissoniennes et un service géométrique. Le temps d'attente dans la queue pour la ième destination est:

$$W_i = \frac{\lambda_i (2 - \Theta)}{2(\Theta - \lambda_i)\Theta} \quad \text{unités de temps de service}$$

Quant au temps d'attente D_i dans le système, il est représenté sur la figure 2.5.

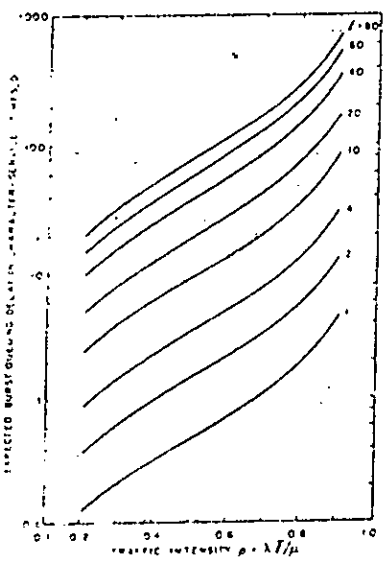


Figure 2.5. Temps d'attente moyen d'une trame dans la queue en fonction de l'intensité du trafic.

3.- MODELE A ENTREE MIXTE ET SERVICE CONSTANT [48]

Dans certains systèmes de communication, le trafic d'entrée est un mélange de trames (chaînes de caractères) et de simples caractères. C'est ainsi, qu'en 1972, une étude a été réalisée par W.W. Chu et L.C. Liang sur les modèles à entrée mixte et service constant, pour déterminer la relation entre la probabilité de débordement et la taille du tampon pour différentes intensités du trafic et le temps d'attente du à la bufferisation pour ce type de modèle.

Le temps d'émission d'un caractère sur la voie multiplexée, est considéré comme temps de service unitaire. L'entrée des caractères x est supposée poissonienne, avec un taux λ_c caractères par unité de service:

$$f_x(k) = \frac{\lambda_c^{\,k}}{k!} \exp(-\lambda_c) \qquad k = 0,1,2,\ldots$$

Quant au trafic d'entrée des trames, on suppose la longueur de la trame y géométriquement distribuée, avec une moyenne $l=1/\Theta$, et le nombre de trames arrivant dans le tampon, durant une unité de service, poissonien avec un taux λ_t trames par unité de service. La distribution de l est:

$$f_y(l) = \Theta(1 - \Theta)^{l-1} \qquad l = 1,2,\ldots$$

et la distribution du nombre de trames arrivant durant une unité de service est:

$$f_z(n) = \frac{\lambda_t^{\,n}}{n!} \exp(-\lambda_t) \qquad n = 0,1,2,\ldots$$

La probabilité de débordement P_{0f} est égale à:

$$P_{0f} = 1 - \alpha/\tau$$

avec:

$$\tau = \lambda_c + \lambda_t l$$
$$\alpha = 1 - p_0 \quad (p_0 = \text{probabilité de tampon vide})$$

Lorsque cette probabilité de débordement est très petite, une bonne approximation du temps d'attente moyen D d'une trame dans le système, peut être obtenue à partir du même système de file d'attente à taille infinie, soit:

$$D = \frac{\rho}{2(1-\rho)} + \frac{\alpha(1-1)\rho}{1-\rho} \quad \text{unités de services}$$

où: $\rho = \lambda_c + 1\lambda_t$
$\alpha = \lambda_t 1/\rho = 1 - \lambda_c/\rho$

Ce temps d'attente est représenté, en fonction de la longueur moyenne de la trame et de l'intensité du trafic, sur la figure 2.6.

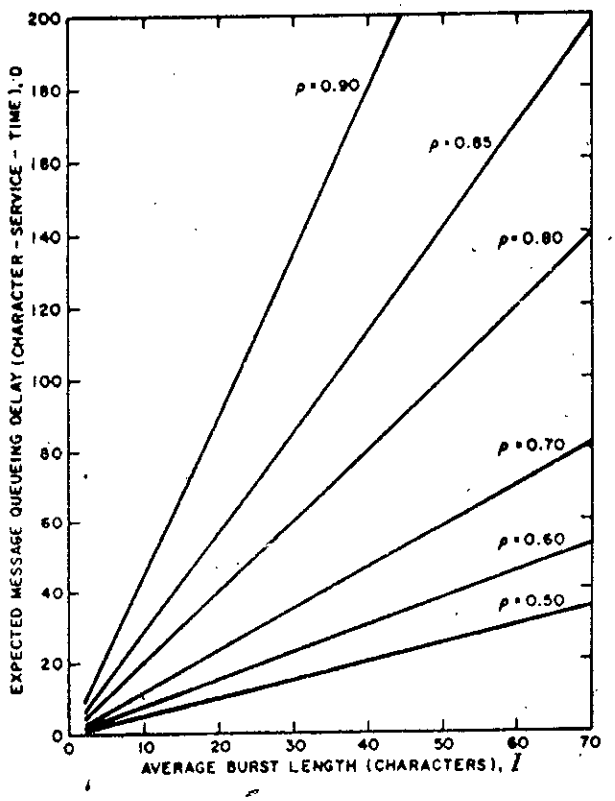


Figure 2.6. Temps d'attente moyen d'une trame dans la queue en fonction de la longueur moyenne de la trame pour une intensité de trafic mixte de $\alpha = 0.5$.

4.- MODELE A PRIORITES MULTIPLES [50]

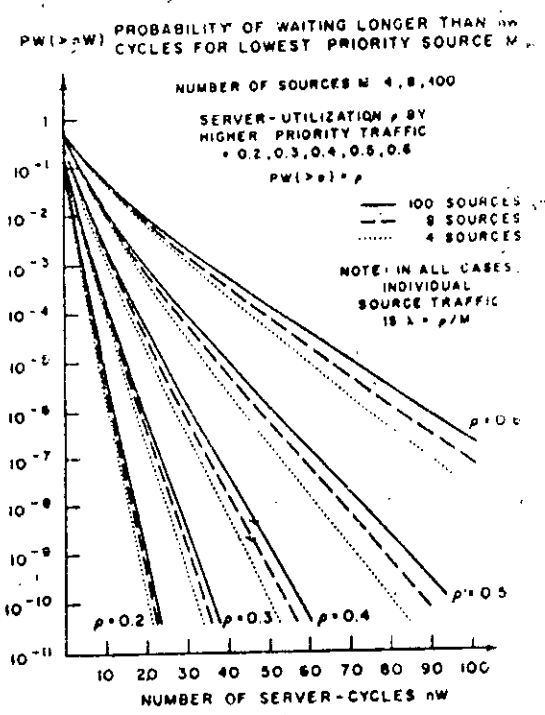
Dans un système de communication, la demande de service n'est pas toujours équivalente chez les usagers. Les traitements de certains utilisateurs peuvent être plus importants, plus fréquents ou plus impératifs que d'autres. C'est pourquoi, une grande flexibilité dans l'organisation de la communication et l'allocation des voies est requise pour la gestion de ces systèmes.

Le modèle présenté dans ce paragraphe, a été étudié par D.A. Gall et H.R. Mueller en 1972 et est essentiellement caractérisé par la variation des niveaux de priorité attribués au usagers. Il comprendra un nombre de sources finies groupées en processus binomiaux, un tampon de taille finie et sera à service synchrone et constant. Les sources seront servies selon l'ordre de priorité établi. A la fin d'un service la scrutation repart de la dernière source servie et non de la source suivante, comme c'est le cas dans la méthode du «round-robin».

Plusieurs expériences ont été réalisées pour cette étude. La première se penche sur l'influence de la grandeur des classes de sources sur le temps d'attente des messages de la source la moins prioritaire. Le trafic est supposé être égal, sur les différentes sources. Les résultats fournis par des tests menés sur des classes de 4, 8 et 100 sources (les sources de ces classes étant de priorités différentes) montrent que l'influence de la grandeur de la population des classes n'est pas significative. De plus, les résultats de la population des 100 sources se rapprochent étroitement de ceux d'un processus à arrivées poissoniennes et dont le nombre de sources est infini.

La deuxième expérience étudie la réaction de classes de sources à trafic différent, sur le temps d'attente de la source la moins prioritaire. Pour ce faire, une classe de 100 sources a été subdivisée en 5 classes de 55, 24, 12, 6 et 3 sources, les priorités des sources d'une même classe étant égales et la moyenne du trafic de ces sources étant de 0.5. Là encore, les résultats obtenus montrent que la variation du trafic n'agit que très faiblement sur l'attente de l'élément le moins prioritaire.

Il est par ailleurs clair, qu'en raison de l'intensité du trafic sur les sources de haute priorité, le temps d'attente des sources les moins prioritaires est rallongé. Le nombre de cycles



Waiting-time probability distribution $PW(>nW)$ for source M (constant service time). Parameter—higher priority traffic ρ.

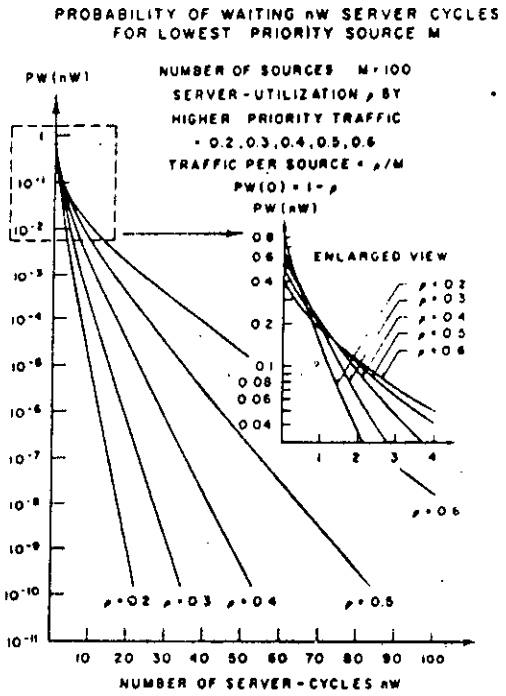


Fig. 5. Waiting-time probability distribution $PW(nW)$ for source M (constant service time). Parameter—higher priority traffic ρ.

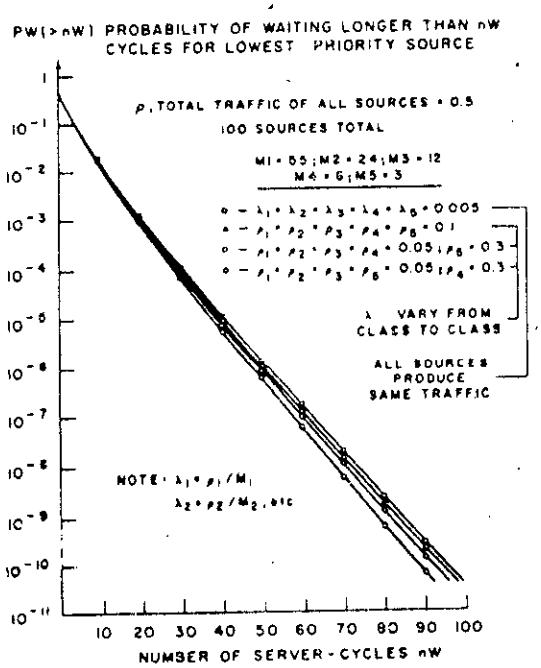


Fig. 7. Influence of terminal traffic mix on waiting-times $PW(nW)$ (constant service time). Parameter—higher priority traffic ρ.

moyen que la source la moins prioritaire doit attendre est (service compris):

$$\overline{nW} = +1 \sum_{nW=0}^{\infty} nW \; PW(nW)$$

Un calcul de la moyenne de la file d'attente de ce processus pour un tampon de longueur finie a été effectué à partir de la distribution de la longueur de la file, et comparé à la moyenne d'une file dont le tampon est de longueur infinie, déterminée par Silk [43],

$$\overline{nQ} = \frac{\rho}{1-\rho} [1 - (\rho/2)(1+1/m)]$$

Cette formule suppose un processus à entrée binomiale de m sources et un trafic de $\rho = \lambda m$. L'erreur entre la formule de Silk et la queue virtuelle finie est de 10^{-14} pour NQ grand.

5.- MODELE A ENTREE BINOMIALE ET SERVICE CONSTANT [52]

Les processus poissoniens sont basés sur la supposition d'un nombre infini de sources ou de terminaux. Rudin [47] trouva plus convenable de supposer les arrivées binomiales, lorsque le nombre des terminaux est petit et calcula les probabilités de débordement et la moyenne des temps d'attente pour un modèle à plusieurs serveurs.

Le modèle étudié dans ce paragraphe fut élaboré par G.F.W. Fredrikson en 1974 [52]. Il sera à entrée binomiale, service constant, longueur de trame constante, serveur unique et taille du tampon illimitée. Le nombre des sources d'entrée est N, et la vitesse de transmission sur chacune de ces entrées est r_1 message par unité de temps. La vitesse de la voie composite est r_2 ; et pour assurer une forte utilisation de cette voie, on suppose:

$$N\ r_1 > r_2$$

Le temps requis pour la transmission d'une trame de taille constante sur la voie composite est:

$$t_0 = 1/r_2$$

Quant au nombre maximum de trames arrivant pendant l'intervalle de temps t_0, est:

$$n = \frac{Nr_1}{r_2}$$

Avec une entrée binomiale, la probabilité de j arrivées pendant le temps t_0 est:

$$k_j = \binom{n}{j} \beta^j (1-\beta)^{n-j} \qquad j = 0,1,2,\ldots,n$$

$$k_j = 0 \qquad j < 0, j > n$$

Le processus d'arrivée décrit par le système ci-dessus, n'est strictement correct que lorsque la réception des messages répond aux conditions suivantes:

— les trames arrivent dans le tampon à des temps discrets séparés par un intervalle de temps $dt = t_0/n$;
— une seule trame peut arriver en ce temps;
— l'arrivée d'une trame est un événement statistique indépendant des arrivées précédentes.

le nombre moyen des arrivées durant le temps t_0 (ou trafic) est:

$$\rho = \sum_{j=0}^{n} jk_j = n\beta$$

Quant à la longueur moyenne de la file d'attente, elle est établie par la formule suivante:

$$E\{Q\} = n\beta + \frac{n(n-1)\beta^2}{2(1-n\beta)}$$

Cette expression est approximativement correcte, pour un processus dont le tampon est de taille finie, et dont la probabilité de débordement est petite.

Par ailleurs, sachant que le nombre moyen des arrivées durant le temps d'attente moyen, $E\{w\}$, plus le temps de service, t_0, doit être égal à la longueur moyenne de la file d'attente, $E\{Q\}$:

$$(E\{w\} + t_0)\alpha = E\{Q\}$$

où α est le nombre moyen des arrivées par unité de temps. Et sachant que la probabilité d'arrivée d'une trame à des temps fixes séparés par $dt = t_0/n$, est β, donc:

$$\alpha = \frac{\beta}{dt} = \frac{n\beta}{t_0}$$

le temps d'attente moyen, exprimé en temps de service unitaire,

sera:

$$\frac{E\{w\}}{t_0} = \frac{(n-1)\beta}{2(1-n\beta)}$$

Les résultats numériques fournis par ce processus (figure 2.7) révèlent que l'utilisation d'une entrée binomiale avec des petites valeurs de n présente de très faibles probabilités de débordement.

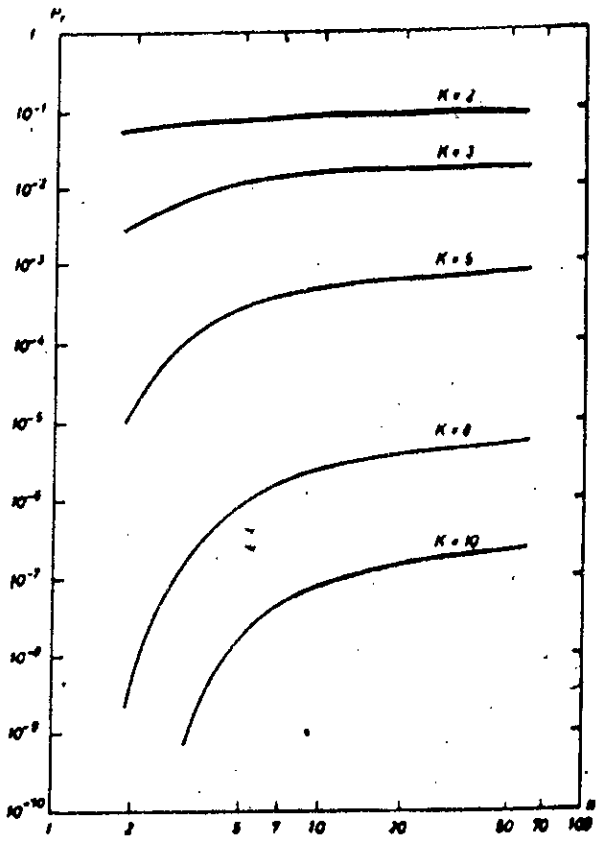


Figure 2.7. Probabilité de rejet d'une trame en fonction du nombre maximum d'arrivées durant un intervalle de service; $\rho = 0.4$.

6.- MODELE A RETRANSMISSIONS ARQ [53]

Différentes méthodes de détection d'erreurs et de retransmission (ARQ) sont utilisées pour le contrôle des erreurs générées lors d'une communication entre deux multiplexeurs statistiques à travers une ligne bruitée. Dans certaines méthodes, la trame reste stockée dans le tampon du multiplexeur émetteur, jusqu'à ce que ce dernier soit informé (à l'aide d'un message ACK/NACK) que la trame a été correctement reçue par le multiplexeur destinataire. Ce stockage de trames, après leur émission, affecte la longueur des files d'attente et le temps d'attente des trames du multiplexeur.

Dans cet article, réalisé par D. Towsley et J.K. Wolf en 1979 [53], deux stratégies ARQ sont utilisées: une ARQ à arrêt-et-attente et une ARQ continue. Pour chaque stratégie, un contrôle du bloc reçu est effectué. Si la trame est correctement reçue, un message ACK est envoyé à la station émettrice et une nouvelle trame peut être émise. Si, par contre, le message est incorrectement reçu, un message NACK est transmis à la station émettrice qui renvoie la même trame. Dans le système arrêt-et-attente, après chaque transmission de trame, l'émetteur est inactif, en attente d'un message ACK/NACK du récepteur. La stratégie de transmission continue réduit ces temps d'inactivité en transmettant les trames de manière continue, sans attendre l'accusé de réception du récepteur. Lorsqu'un message NACK arrive pour une trame quelconque, elle est renvoyée ainsi que toutes celles qui la suivent

Le temps sera subdivisé en intervalles de temps unitaires. Le nombre de trames D_j arrivant pendant un intervalle de temps unitaire est une variable aléatoire i.i.d. dont la distribution est de moyenne μ_D et de variance σ_D. Le tampon de réception sera à capacité illimitée. Comme la transmission d'une trame s'écoule (1+s) intervalles de temps, un accusé de réception est renvoyé après s intervalles de temps. Si N_i est le nombre de transmissions d'une trame, M_i sera le nombre d'intervalles de temps total requis à la transmission d'une trame correcte, $M_i = (1+s)N_i$. M_i est une variable aléatoire distribuée selon une loi de probabilité de moyenne μ_M et de variance σ_M. p sera la probabilité d'erreur de transmission sur le canal à l'aller (erreur sur les données), et p' la probabilité d'erreur de transmission au retour (sur les messages ACK/NACK).

Ainsi, le temps d'attente moyen dans la queue, dans un système

à stratégie ARQ à arrêt-et-attente est:

$$W = \frac{\mu_M}{2} + \frac{\mu_D^2 \sigma_M^2 + \mu_M \sigma_D^2}{2\mu_D (1-\mu_D \mu_M)}$$

où:

$\mu_M = (1+s)[(1-p)^{-1} + p'(1-p')^{-1}]$

$\sigma_M^2 = (1+s)^2 [p(1-p)^{-2} + p'(1-p')^{-2}]$

$\mu_D = \sigma_D = \lambda_0$ pour un processus à arrivées poissoniennes de paramètre λ_0.

A l'aide de la formule de Little, on obtient la longueur moyenne de la file d'attente:

$$L = \frac{\mu_M \mu_D}{2} + \frac{\mu_D^2 \sigma_M^2 + \mu_M \sigma_D^2}{2(1-\mu_D \mu_M)}$$

Quant au temps d'attente moyen pour une stratégie ARQ continue, il est égal à:

$$W = \frac{\mu_M + 2s}{2} + \frac{\mu_D^2 \sigma_M^2 + \mu_M \sigma_D^2}{2\mu_D (1-\mu_D \mu_M)}$$

où:

$\mu_M = (1+s)p(1-p)^{-1} + (1+sp')(1-p')^{-1}$

$\sigma_M^2 = (1+s)^2 [p(1-p)^{-2} + p'(1-p')^{-2}]$

$\mu_D = \sigma_D = \lambda_0$ pour un processus à arrivées poissoniennes de paramètre λ_0.

La longueur moyenne de la file d'attente qui en découle, par la formule de Little est:

$$L = \frac{\mu_D (\mu_M + 2s)}{2} + \frac{\mu_D^2 \sigma_M^2 + \mu_M \sigma_D^2}{2(1-\mu_D \mu_M)}$$

Ces grandeurs sont traduites sous forme de graphes [51, pp. 699-702] illustrant l'allure du temps d'attente de la trame et de la longueur de la file d'attente du système en fonction de la probabilité d'erreur aussi bien dans la stratégie à arrêt-et-attente que celle continue.

7.- MODELE A ENTREES CORRELEES [61]

Les modèles étudiés jusqu'à présent font tous, abstraction de la corrélation des entrées. Ce n'est qu'en 1988 que H. Bruneel [61] publiait un article qui traitait des modèles à entrées corrélées, sources indépendantes et finies (m sources), tampon de taille infinie et serveur unique.

Dans ce modèle, du côté des sources (ou utilisateurs), le temps est divisé en «périodes actives» et «périodes passives» selon que l'utilisateur génère des données ou non. Pour chaque utilisateur, ces deux périodes seront supposées être des variables aléatoires i.i.d et de distribution géométrique, comme suit:

- Prob[la période active contient n fractions] = $(1-\alpha)\alpha^{n-1}$ $\quad n \geq 1$
- Prob[la période passive contient n fractions] = $(1-\beta)\beta^{n-1}$ $\quad n \geq 1$

avec $\alpha + \beta \neq 1$;

Afin de bien caractériser le comportement des utilisateurs actifs, on suppose que durant une «fraction de période active» un utilisateur génère un nombre entier positif de paquets avec une fonction de probabilité f(z). Un choix approprié de f(z), permet une description quantitative du degré d'activité des usagers. Pour ce faire, on suppose que les usagers actifs génèrent un message, par fraction de temps active (nombre de paquets égal à la longueur du message); et ces messages étant de longueur différente, on leur suppose une distribution géométrique. On obtient alors:

$$f(z) = \frac{(1-v)z}{1-vz}$$

où $\bar{f} = 1/(1-v)$ est la longueur moyenne des messages. Ainsi, la longueur moyenne de la file sera donnée par:

$$\bar{N} = \frac{m\sigma}{2(1-v)(1-v-m\sigma)}\,[2(1-v)-(3m-1)\sigma+2K(v+(m-1)\sigma)]$$

où $\quad \sigma = \dfrac{1-\beta}{2-\alpha-\beta}$

La figure 2.8 montre le comportement de la longueur moyenne de la file en fonction de l'activité moyenne de l'usager σ et pour différentes valeurs de K. Le paramètre m est choisi égal à 5 usagers et f égal à 2 paquets. Les résultats de la figure illustrent clairement l'influence du paramètre K sur la longueur moyenne de la file d'attente, pour toutes les valeurs de σ. On note que la courbe relative à K=1 correspond au cas où l'entrée des usagers n'est pas corrélée.

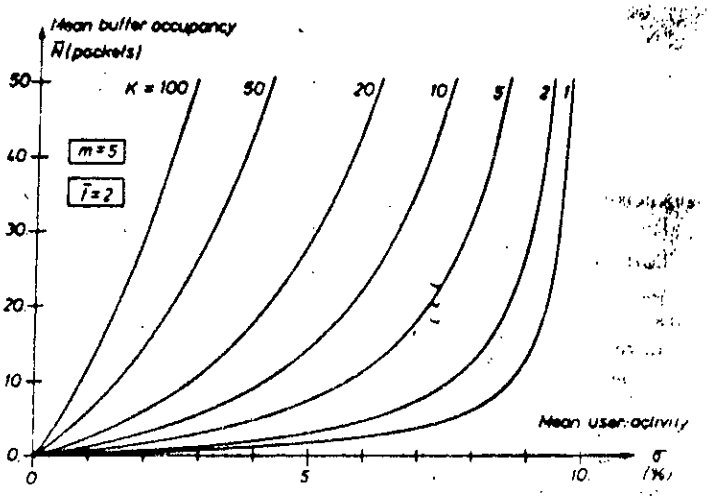


Figure 2.8. Longueur moyenne de la file en fonction de l'activité moyenne de l'usager.

8.- MODELE A TAMPONS SEPARES [57]

Contrairement aux articles précédents qui fondaient leur étude sur un tampon commun aux différentes stations, celui-ci traite le cas de tampons séparés pour ces stations (un tampon par station). Le modèle fut proposé par O.J. Boxma et W.P. Groenendijk en 1988 [57]. Il sera à serveur unique S et à multiples files d'attente (Q1, Q2, ... QN), à service cyclique. Le temps de transit d'une queue à une autre sera, par conséquent, supposé non nul. La stratégie de service des différentes queues peut être variable.

N stations seront donc considérées, chaque station étant pourvue d'un tampon de capacité infinie. Chaque trame du tampon sera constituée de paquets de taille fixe. Le temps sera subdivisé en intervalles égaux au temps de transmission de la donnée contenue dans le paquet (temps de service du paquet).

Soit:

$x_i(j)$ = nombre de trames arrivant à la station i pendant le $j^{ème}$ intervalle de temps,
b_i = nombre de paquets d'une trame de la station i,
t_i = temps de transit entre les stations i et i+1.

Les arrivées sur une station sont supposées être indépendantes de celles des autres stations. Les arrivées (trames) sur la station i, $x_i(j)$, j=1,2,... sont iid selon la loi de Bernoulli, avec:

$$\lambda_i = E[x_i(j)] \qquad \lambda_i^{(2)} = E[x_i^2(j)]$$

$$\lambda = \sum_{i=1}^{N} \lambda_i \qquad \lambda^{(2)} = E\left[\left(\sum_{i=1}^{N} x_i(j)\right)^2\right]$$

$$\beta_i = E[b_i] \qquad \beta_i^{(2)} = E[b_i^2]$$

$$\beta = \sum_{i=1}^{N} \frac{\lambda_i}{\lambda} \beta_i \qquad \beta^{(2)} = \sum_{i=1}^{N} \frac{\lambda_i}{\lambda} \beta_i^{(2)}$$

$$\rho_i = \lambda_i \beta_i \ , \ i=1,2,\ldots,N \qquad \rho = \sum_{i=1}^{N} \rho_i$$

Les temps de transit t_i sont des variables aléatoires iid,

avec:

$$s_i = E[t_i] \qquad s_i^2 = E[t_i^2]$$

$$s = \sum_{i=1}^{N} s_i$$

La stratégie de service, qui peut varier d'une station à une autre, entend le nombre de messages servis au moment où le serveur S scrute la station. Elle peut être:

— service exhaustif: S sert Q_i jusqu'à son vidage,
— service régulé: S sert juste les trames présentes dans Q_i au moment de son arrivée,
— service limité: S sert juste une trame de Q_i,
— service semiexhaustif: S poursuit le service des trames de Q_i jusqu'à ce que leur nombre soit inférieur à celui qu'il a trouvé à son arrivée.

Ainsi, la somme pondérée des temps d'attente moyens d'une trame s'écrit comme suit:

$$\sum_{i=1}^{N} \rho_i EW_i = \frac{\lambda\beta^{(2)}}{2(1-\rho)}\rho + \frac{(\lambda^{(2)} - \lambda^2 - \lambda)\beta}{2\lambda(1-\rho)}\rho + \rho\frac{s^{(2)}}{2s} - \frac{1}{2}\rho$$

$$+ \frac{s}{2(1-\rho)}(\rho^2 - \sum_{i=1}^{N}\rho_i^2) + \sum_{i=1}^{N} EM_i$$

où EM_i dépend du choix de la stratégie de service:

— service exhaustif (noté e)

$$EM_i = 0$$

— service régulé (noté r)

$$EM_i = \rho_i^2 \frac{s}{1-\rho}$$

— service limité (noté l)

$$EM_i = \frac{\lambda_i s}{1-\rho} \rho_i EW_i + \rho_i^2 \frac{s}{1-\rho} + \frac{\rho_i s}{1-\rho} \frac{\lambda_i^{(2)} - \lambda_i}{2\lambda_i}$$

— service semiexhaustif (noté se)

$$EM_i = \rho_i \frac{\lambda_i s(1-\rho_i)}{1-\rho} EW_i - \frac{\lambda_i s(1-\rho_i)}{1-\rho} \left[\frac{\lambda_i \beta_i^{(2)}}{2(1-\rho_i)} \rho_i + \frac{(\lambda_i^{(2)} - \lambda_i^2 - \lambda_i)}{2\lambda_i (1-\rho_i)} \rho_i \beta_i \right]$$

Pour un service cyclique comprenant les différentes stratégies, nous obtenons: .

$$\sum_{i\varepsilon e} \rho_i EW_i + \sum_{i\varepsilon r} \rho_i EW_i + \sum_{i\varepsilon l} \rho_i \left[1 - \frac{\lambda_i s}{1-\rho}\right] EW_i + \sum_{i\varepsilon se} \rho_i \left[1 - \frac{\lambda_i s(1-\rho_i)}{1-\rho}\right] EW_i$$

$$= \frac{\lambda \beta^{(2)}}{2(1-\rho)} \rho + \frac{(\lambda_i^{(2)} - \lambda_i^2 - \lambda_i)\beta}{2\lambda(1-\rho)} \rho + \rho \frac{s^{(2)}}{2s} - \frac{1}{2} \rho$$

$$+ \frac{s}{2(1-\rho)} \left[\rho^2 - \sum_{vi} \rho_i^2\right] + \frac{s}{(1-\rho)} \sum_{i\varepsilon r,l} \rho_i^2 - \frac{s}{2(1-\rho)} \sum_{i\varepsilon se} \lambda_i^2 \beta_i^{(2)} \rho_i$$

$$+ \frac{s}{1-\rho} \sum_{i\varepsilon l} \frac{\lambda_i^{(2)} - \lambda_i}{2\lambda_i} \rho_i - \frac{s}{2(1-\rho)} \sum_{i\varepsilon se} (\lambda_i^{(2)} - \lambda_i^2 - \lambda_i)\beta_i \rho_i$$

Si l'on suppose les arrivées poissoniennes et l'on prend donc $\lambda_i^{(2)} = \lambda_i^2 + \lambda_i$, on obtient la relation suivante pour la somme pondérée des temps d'attente moyens:

$$\sum_{i\varepsilon e} \rho_i EW_i + \sum_{i\varepsilon r} \rho_i EW_i + \sum_{i\varepsilon l} \rho_i \left[1 - \frac{\lambda_i s}{1-\rho}\right] EW_i + \sum_{i\varepsilon se} \rho_i \left[1 - \frac{\lambda_i s(1-\rho_i)}{1-\rho}\right] EW_i$$

$$= \frac{\lambda \beta^{(2)}}{2(1-\rho)} \rho + \rho \frac{s^{(2)}}{2s} - \frac{1}{2} \rho + \frac{s}{2(1-\rho)} \left[\rho^2 - \sum_{vi} \rho_i^2\right] + \frac{s}{(1-\rho)} \sum_{i\varepsilon g,l} \rho_i^2$$

$$- \frac{s}{2(1-\rho)} \sum_{i\varepsilon se} \lambda_i^2 \beta_i^{(2)} \rho_i + \frac{s}{2(1-\rho)} \sum_{i\varepsilon l} \lambda_i \rho_i$$

9.- MODELE A TAMPONS REDUITS [60, 64]

Cet article se distingue de ses prédécesseurs par la taille des tampons de stockage de l'information échangée, ces tampons étant séparés pour les différentes stations de travail (un tampon par station). Il a été élaboré en 1988 par T. Takine, Y. Takahashi et T. Hasegawa [60], puis complété par B. Mukherjee, C.K. Kwok, A.C. Lantz et W-H.L.M. Moh en 1990 [64]. Deux variantes du modèle seront considérées: une variante pour le système conventionnel et une autre pour le système relaxé. Dans le système conventionnel, le tampon a une taille égale à celle de la trame. Par conséquent, une nouvelle trame ne peut accéder au tampon que lorsque la précédente est entièrement émise, celles arrivant entre temps étant perdues. Cette contrainte est levée dans le système relaxé, où une nouvelle trame peut pénétrer le tampon tandis que la précédente est en cours d'émission, la taille de ce tampon étant le double de celle de la trame. Ce modèle suppose une communication en mode interactif, où une nouvelle trame n'est générée que lorsque la précédente est entièrement émise [54, 55].

Dans les deux systèmes, les M stations sont sondées de manière cyclique (ou bouclée). Si une trame est disponible au moment du sondage, elle est transmise sur la voie composite, sinon la station suivante est testée. Les arrivées sur une station i sont supposées être indépendantes et poissoniennes de paramètre λ_i. Le temps de transmission h_i, d'un message de la station i, obéit à une fonction de distribution de probabilité générale $H_i(t)$, de moyenne H_i. Le temps de transition de la station i à la station i+1, est noté u_{i+1} pour le système conventionnel et u_i pour le système relaxé. u_i est supposé indépendamment distribué relativement à la fonction de distribution de probabilité générale $U_i(t)$, dont la moyenne est U_i.

Ainsi, le nombre moyen de trames transmises par la station k, Θ_k, par unité de temps, est donné par:

$$\Theta_k = \frac{\alpha_k}{C}$$

où:

— α_k (k=1,...,M) est la probabilité de disponibilité d'une trame à la station k, au moment du sondage, et

— C est la longueur moyenne du temps cyclique, ou intervalle de temps entre deux sondages consécutifs d'une même station, tel que:

$$C = \sum_{k=1}^{M} \{U_k + \alpha_k H_k\}$$

Quant aux temps d'attente moyens dans la queue W_k, et dans le système D_k, ils sont donnés comme suit:

$$W_k = 1/\Theta_k - 1/\lambda_k - H_k$$
$$D_k = 1/\Theta_k - 1/\lambda_k$$

La figure 2.9 montre l'attente moyenne d'une trame en deux stations distinctes, pour les deux types de systèmes, en fonction de la charge totale. Cette charge est égale à la somme des taux d'arrivée de toutes les stations. La première station représentée (station 1) se distingue par une forte intensité du trafic (charge de 90% de la charge totale). Quant à la deuxième (station 5), elle est de faible intensité. Nous remarquons, sur cette figure, que pour les faibles valeurs de la charge totale, le temps d'attente dans la station fortement chargée est inférieur à celui de la station à faible charge, et ce d'autant plus dans le système conventionnel que le système relaxé. Pour les grandes valeurs de la charge totale, la station chargée atteint la saturation (α_k tend vers 1) bien plus tôt que celle à moindre charge.

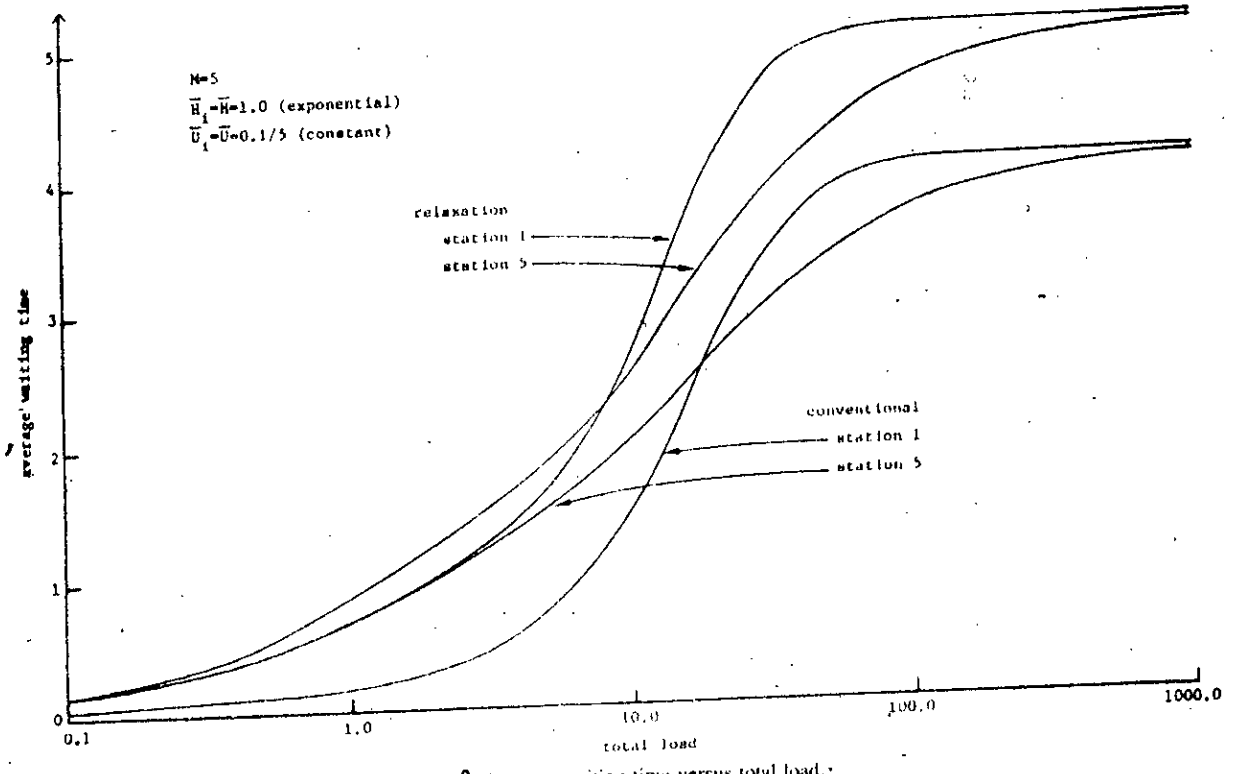


Fig. 2.9. Average waiting time versus total load.

10.- CONCLUSION

Après exposé de tous ces modèles, il apparaît clairement que jusqu'en 1987 l'étude analytique des multiplexeurs statistiques était basée sur la considération d'un tampon unique pour les différentes stations de communication d'un même site; or du point de vue pratique, ceci ne constitue pas la meilleure structure de données pour une gestion optimale de la communication, du fait de la lenteur qui découle de la tâche requise pour la gestion du nombre élevé de pointeurs, ce nombre étant imposé par le choix de la structure du tampon unique. Ce n'est qu'en 1988 que la structure des tampons séparés fut modélisée, mais sans apporter de réponse aux préoccupations des spécialistes de la matière, généralement soucieux de déterminer les deux paramètres classiques, à savoir, la longueur de la file d'attente et le temps d'attente moyen dans la file (seule la somme pondérée des temps d'attente moyens d'une trame dans la file, fut déterminée).

De plus, les différentes spécificités des modèles sont toujours traitées de manière séparée (modèle à retransmissions ARQ ou modèle à entrées corrélées ou modèle à tampons séparés, etc...), et l'on ne trouve donc pas de modèle qui comprendrait toutes les caractéristiques réunies.

Aussi, afin de bien déterminer les deux paramètres précités et d'autres encore qui contribueraient à l'optimisation de notre produit réel, et pour acquérir une maîtrise totale du produit avant sa réalisation pratique, en prévoyant les situations inattendues et les cas de stress, pour un modèle à différentes caractéristiques, nous avons jugé opportun de simuler notre produit avant sa réalisation pratique. En effet, une étude analytique de notre système s'avérerait très complexe, vue la complexité de ce dernier, mais pourrait, par contre, constituer une extension de ce travail.

CHAPITRE 3

SYSTEME MATERIEL

Les premiers multiplexeurs apparus, notamment ceux dont le multiplexage est fréquentiel, se réduisent à des cartes d'architecture complexe, basées sur la logique cablée. L'apparition des microprocesseurs dans les années 1970 impliqua l'apparition des multiplexeurs temporels statistiques et réduisit de manière considérable la complexité des cartes en introduisant une vision logicielle du problème.

Depuis, la technologique n'a cessé d'évoluer et l'intégration des composants se fait de plus en plus dense. Ainsi, en 1987, un nouvel interface de communication est apparu intégrant huit (8) interfaces de communication asynchrones dans un même boîtier. C'est le SCC2696 de Signetics (OCTAL-UART). Ce composant nous a permis d'optimiser de manière considérable l'architecture de la carte multiplexeur: au lieu de huit (8) composants séparés en raison d'un composant par voie asynchrone un seul est utilisé pour les huit (8) voies.

En outre, le souci des spécialistes d'améliorer les performances des produits élaborés, ne cesse de croître. C'est ainsi que l'OCTAL-UART paraît avec de nouvelles caractéristiques: un registre de réception quadruplement bufferisé et un signal automatique stoppant momentanément le débit de l'information en cas de remplissage de ce registre. De même, vers les années 1985, les interfaces de communication synchrones dotés de protocoles de communication inondèrent le marché, soustrayant au programmeur la gestion de la communication sur la voie composite.

1.- DESCRIPTIF

L'appareil physique que nous avons réalisé comprend:

— Une carte principale réalisée à l'aide du microprocesseur 6809, permettant le multiplexage de huit (8) voies asynchrones à travers

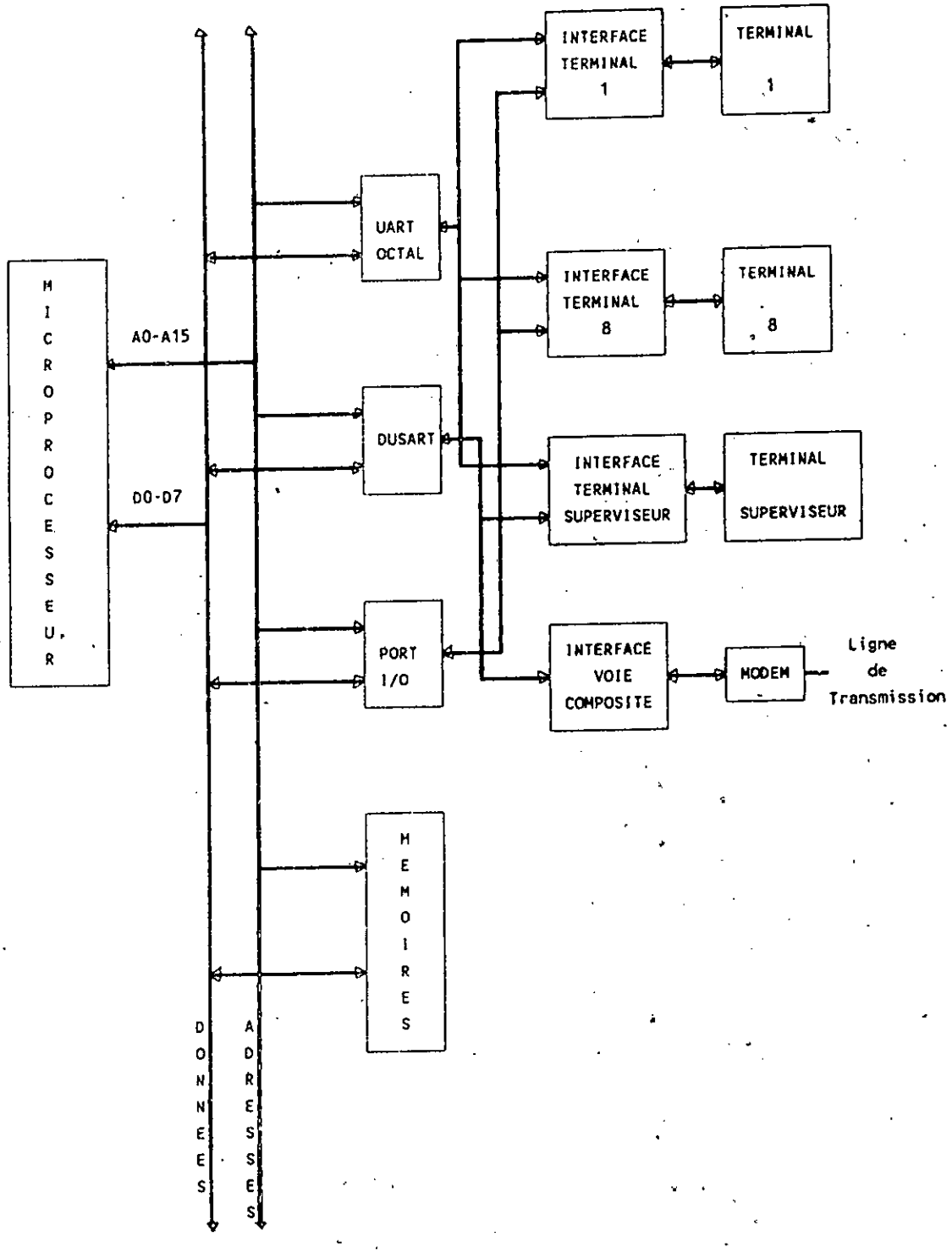


Figure 3.1. Schéma synoptique de la carte principale.

une voie composite synchrone haute vitesse (9600 bauds), sous le contrôle d'une voie de supervision.
— Une carte à l'arrière de l'appareil comprenant les jonctions RS232 des voies basse vitesse, la jonction du superviseur et la jonction de la voie haute vitesse.
— Une carte à l'avant signalant l'état de chaque voie à tout moment, à l'aide de leds.

2.- MICROPROCESSEUR [8]

2.1.- GENERALITES

Le microprocesseur 6809 est particulièrement intéressant dans notre cas, vu qu'un investissement (système de développement) assez important a été consenti pour ce microprocesseur au niveau de la Direction de la Recherche et du Développement de l'Entreprise Nationale des Systèmes Informatiques, aussi bien du côté matériel que logiciel. C'est un 8 bits de haute gamme, dont l'organisation interne est orientée 16 bits. Il peut adresser 64 K-octets par l'intermédiaire de son bus d'adresse. Des modes d'adressage puissants alliés à un jeu d'instructions très performant, bien que le nombre de mnémoniques soit limité, lui donnent d'importantes possibilités logicielles. En effet, le MPU 6809 possède 59 instructions de base mais en conjonction avec les 9 modes d'adressage disponibles, on parvient à 1464 possibilités.

Il possède trois niveaux d'interruptions matérielles (NMI, FIRQ, IRQ), trois niveaux d'interruptions logicielles (SW1, SW2, SW3) et des instructions ayant des rapports avec ce fonctionnement en interruption (CWAY, SYNC, RTI).

2.2.- ORGANISATION MATERIELLE

Le microprocesseur 6809 se présente sous forme d'un boîtier DIL, 40 broches et mono-tension (5V). Son fonctionnement est rythmé par une horloge de 8MHz qui lui permet de fournir une fréquence de 2MHz. Il possède trois bus indépendants:

— un bus de données de 8 bits,
— un bus d'adresses de 16 bits,
— un bus de contrôle de 10 bits.

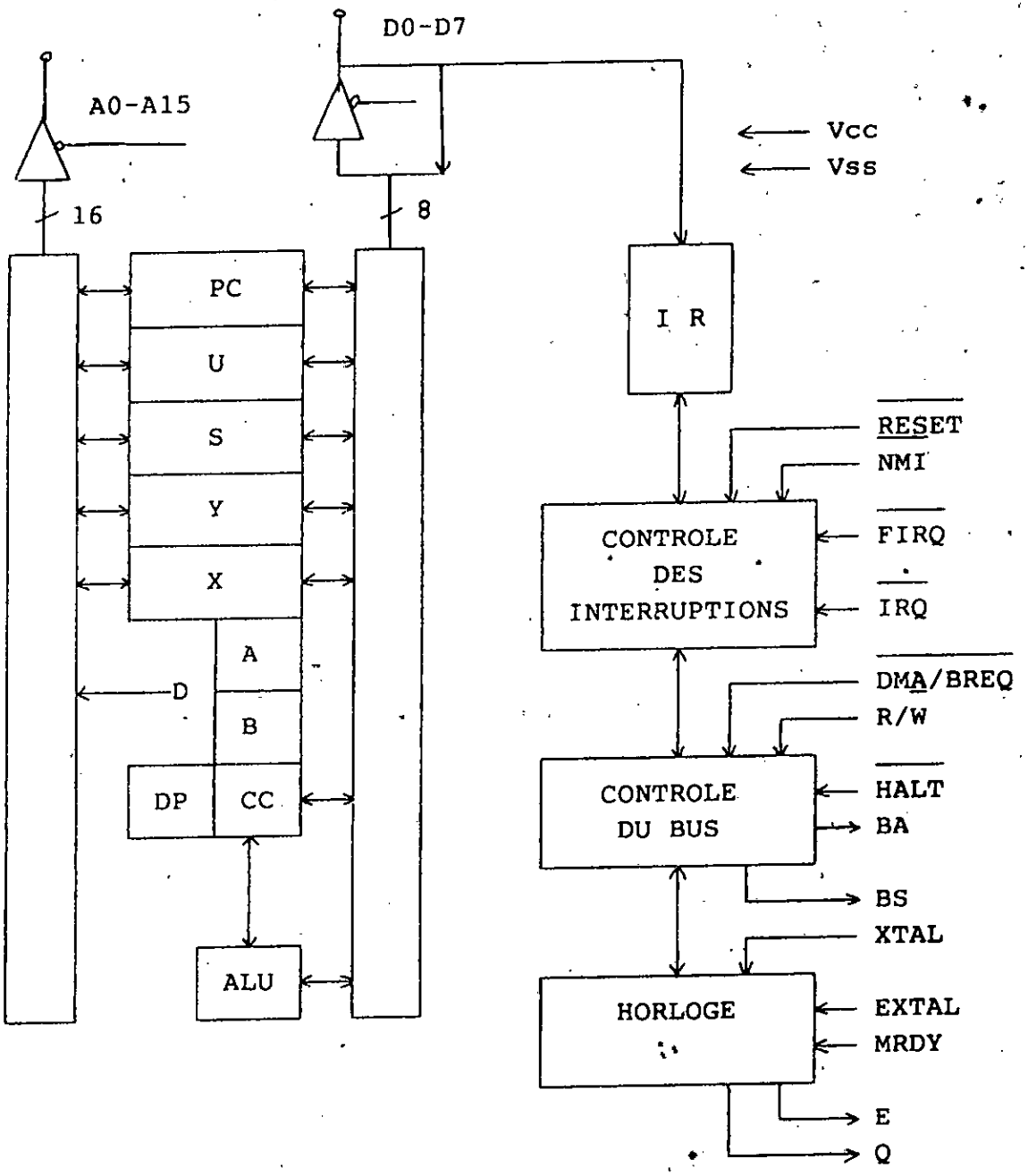


Figure 3.2. Schéma fonctionnel du 6809.

2.2.1.- Signaux horloges

Les entrées horloges sont fournies par les deux broches XTAL et EXTAL à travers lesquelles l'oscillateur interne est connecté à un quartz externe de 8MHz.

Les sorties horloges E out et Q out sont utilisées pour générer un signal de validation de l'accès mémoire. En effet, le signal E est identique au signal d'horloge du système dont la fréquence est celle qui pilote le microprocesseur. Le signal Q, est un signal en quadrature avec E. Les adresses du microprocesseur sont validées sur le front montant du signal Q, et les données sont mémorisées sur le front descendant de E. Le diagramme des temps qui suit montre comment est généré le signal de validation de l'accès à la mémoire VMA.

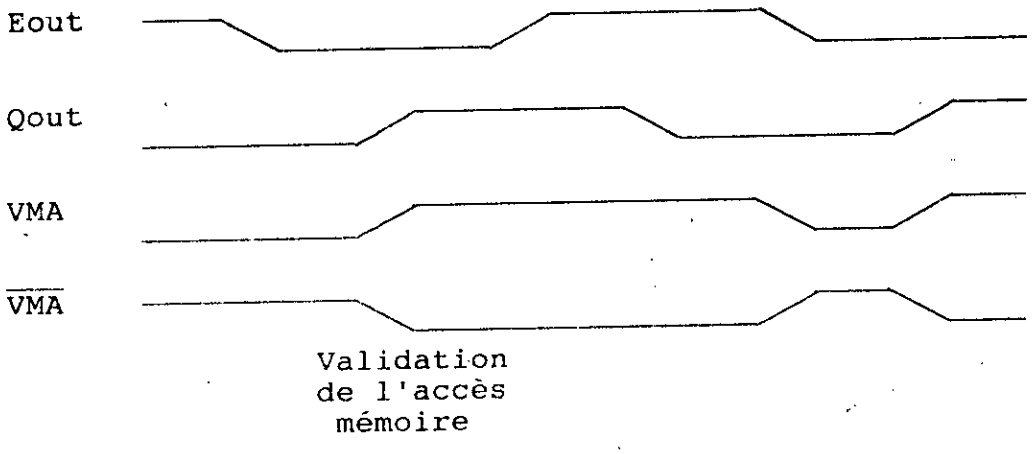


Figure 3.3. Signal de validation de l'accès mémoire

2.2.2.- Signaux de contrôle

Ils comprennent les signaux de:

— **Lecture/Ecriture : R/W**
Ce signal indique le sens du transfert des données sur le bus.

— **Allongement de l'horloge E : MRDY**
Ce signal permet l'allongement de E, pour l'utilisation de mémoires ou de périphériques à accès lent (par exemple, l'interface synchrone/asynchrone à 2 voies).

2.3.- ARCHITECTURE INTERNE

Le microprocesseur 6809 comporte neuf registres internes programmables, accessibles pour l'utilisateur.

— **Accumulateurs : A, B et D**

Le «double accumulateur 16 bits» ou registre D est le résultat de la concaténation des accumulateurs A et B.

— **Registres d'index : X, Y**

Ces deux pointeurs de données (16 bits), parfaitement identiques sont utilisés dans les modes d'adressage indexé.

— **Registres pointeurs de pile : U, S**

Le pointeur de pile S (système) est utilisé par le microprocesseur pour gérer la sauvegarde de ses registres internes, pendant l'exécution de programmes d'interruptions ou de sous-programmes. Le pointeur de pile U (utilisateur) est utilisé uniquement par le programmeur pour réaliser des passages d'arguments de ou vers des sous-programmes. Ils peuvent également être utilisés comme index.

— **Registre compteur programme : PC**

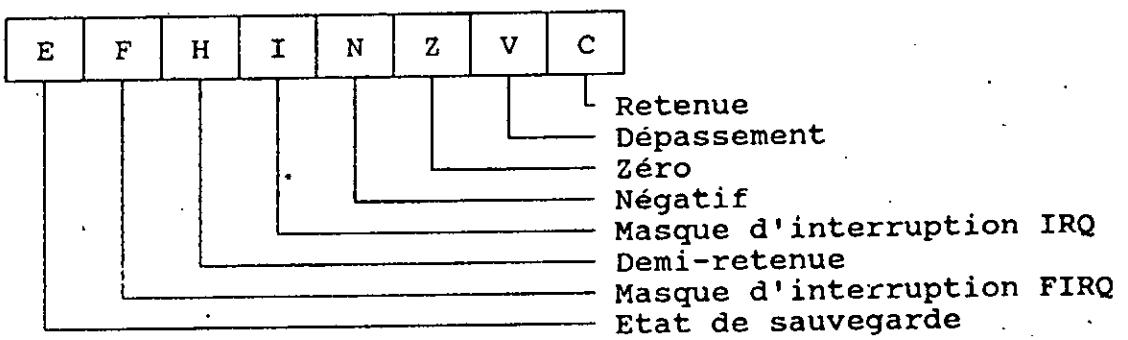
Le contenu de ce registre (16 bits) détermine l'adresse de l'instruction que doit exécuter le microprocesseur.

— **Registre de page : DP**

Ce registre (8 bits) est prévu pour étendre les possibilités d'adressage direct à tout l'espace mémoire. Son contenu concaténé à l'octet suivant le code opératoire d'une instruction en adressage direct, contrôle le bus adresses du microprocesseur.

— **Registre de code condition : CCR**

Ce registre (8 bits) définit à tout instant l'état des indicateurs du processeur: les bits de ce registre sont:



3.- INTERFACE DE COMMUNICATION ASYNCHRONE OCTAL [9]

La communication du côté des voies basse vitesse est assurée par un interface de communication asynchrone octal. Cet interface de communication est aussi appelé OCTAL-UART (Octal Universal Asynchronous Receiver/Transmitter). C'est le SCC2696 de Signetics.

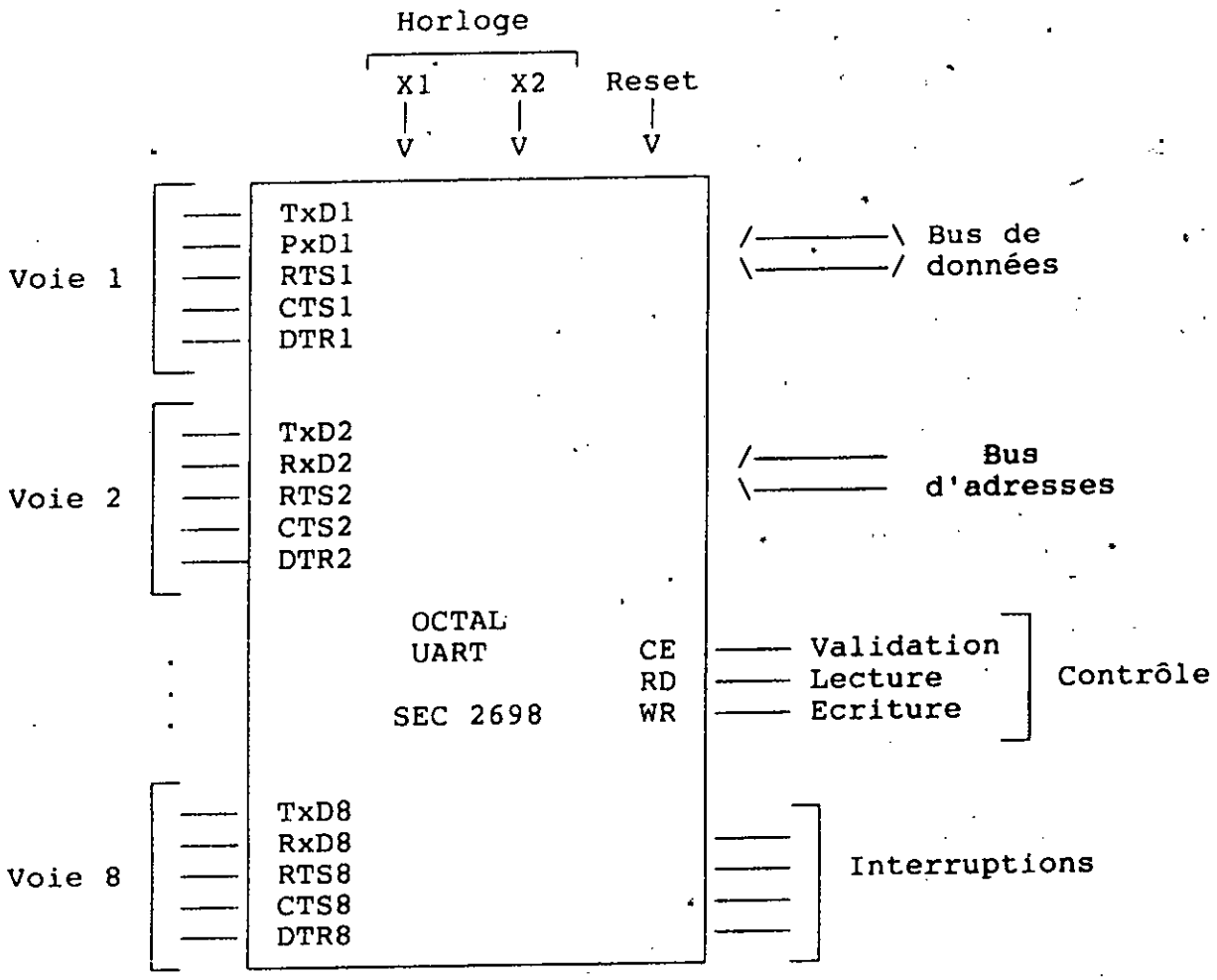


Figure 3.4. Interface voies basse vitesse

L'apparition de ce composant sur le marché en 1987 se fit non sans grands apports pour le domaine du multiplexage:

— la structure de la carte se vit considérablement simplifiée: au lieu de huit boîtiers UART simples, un boîtier UART unique de 64 broches (figure 3.4) comprenant les huit (8) UART intégrés, est utilisé;

— les buffers de lignes (buffers de réception) qui jusqu'aux années 1985/86 étaient doublés seulement, y sont quadruplés (nous les appellerons FIFOs de réceptions);
— le contrôle des FIFOs est automatique: en cas de remplissage de ces FIFOs, le débit sur les voies correspondantes est arrêté par génération de signaux de contrôle de la communication (CTS haut).

Ces huit UART intégrés, seront utilisés pour la communication sur les voies basse vitesse.

L' OCTAl-UART dispose d'un bus de données de huit bits, d'un bus d'adresses de 6 lignes, de trois lignes pour le contrôle de la communication, de signaux pour le contrôle du dialogue entre l'interface et le périphérique, de quatre lignes d'interruption et de deux entrées d'horloge.

Tel que spécifié sur le diagramme bloc (figure 3.6), l'OCTAL-UART comprend: un bus de données amplifié, un contrôle d'interruption, un contrôle d'opérations, une circuiterie d'horloge et huit canaux d'émission/réception. Ces canaux sont regroupés en quatre blocs de deux canaux chacun (figure 3.5).

Bloc A Canaux a,b	Bloc B Canaux c,d
Bloc C Canaux e,f	Bloc D Canaux g,h

Figure 3.5. Architecture des canaux

3.1.- Contrôle d'interruption

Une sortie d'interruption (INTRN) par bloc est prévue pour les situations suivantes:

— registre d'émission prêt,
— registre de réception prêt ou FIFO de réception pleine,
— changement dans l'état du «break»,
— compteur atteint le comptage final,
— changement dans la broche MPI.

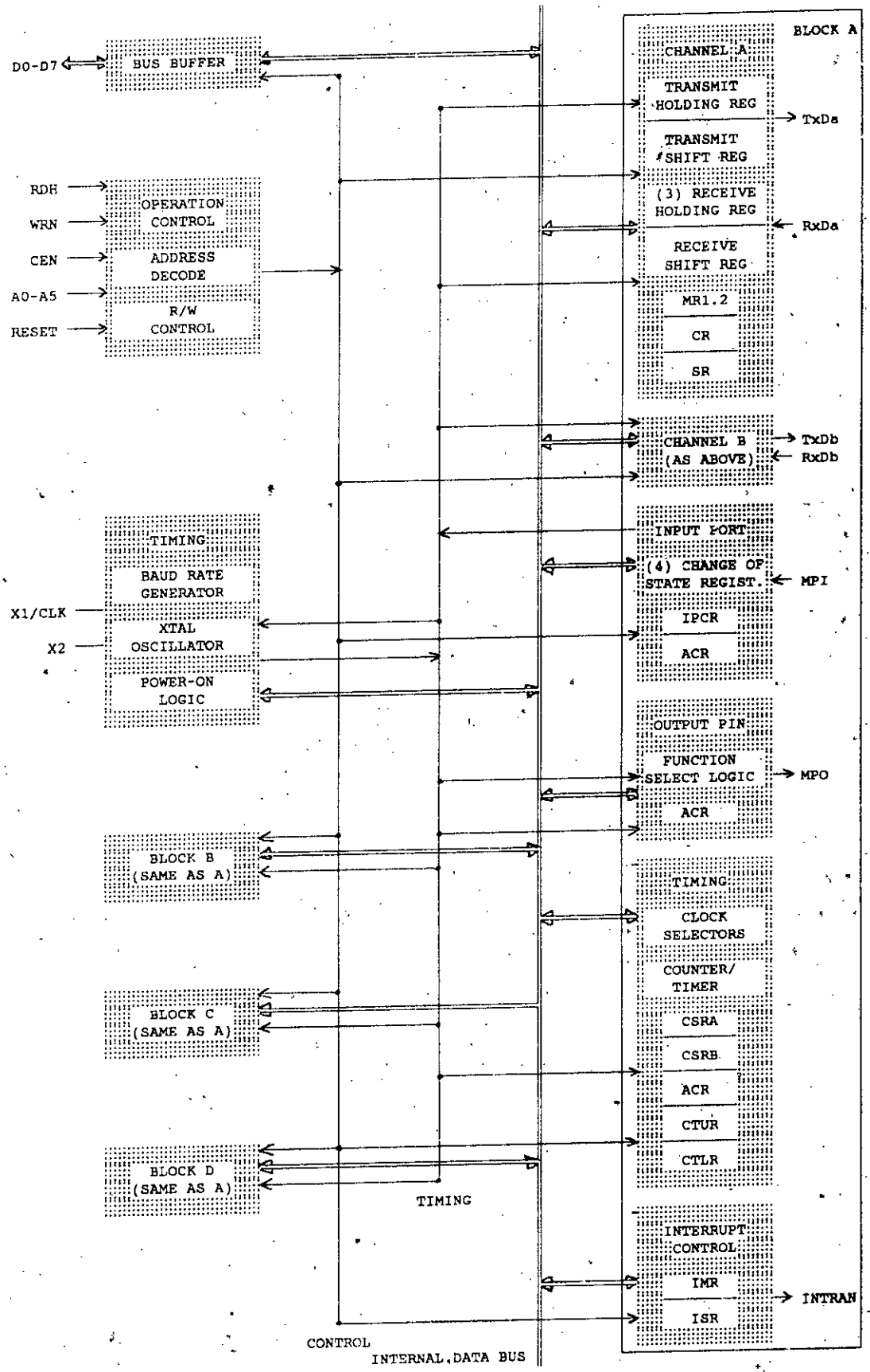


Figure 3.7. Diagramme block de l'UART-OCTAL.

Sont associés à ce système d'interruption les Registres Masque d'Interruption (IMR) et Registres d'Etat d'Interruption (ISR). Le Registre IMR est utilisé pour sélectionner les causes d'interruptions. Le registre ISR détermine les interruptions survenues et non encore traitées.

Le niveau d'interruption auquel sont associées ces causes d'interruptions est l'IRQ. La cause que nous avons retenue est «FIFO de réception pleine».

3.2.- Contrôle d'opération

Ce contrôle concerne celui des opérations de décodage d'adresse et de lecture/écriture.

Les Registres de Mode 1 et 2 (MR1 et MR2) sont accessibles à l'aide d'un pointeur auxiliaire. Le pointeur est positionné sur MR1 après une réinitialisation du système. Toute opération de lecture ou d'écriture sur l'un des deux registres de mode, positionne le pointeur sur MR2. Le pointeur reste sur MR2 jusqu'à ce que la commande Réinitialisation du Pointeur de MR, du Registre de Commandes soit appelée.

3.3.- Circuit d'horloge

Le circuit d'horloge consiste en un oscillateur à quartz, un générateur de vitesse du baud, un compteur/horloge programmable de 16 bits et deux sélecteurs d'horloge.

L'oscillateur à quartz opère à partir d'un quartz de 3.686419 MHz connecté à travers les broches X1/CLK et X2. Il sert de base de temps pour le générateur de vitesse du baud, le compteur/horloge et d'autres circuits internes.

Le générateur de vitesse du baud peut générer dix-huit (18) vitesses différentes, variant entre 50 et 38.4 kbauds, dont treize (13) sont disponibles, simultanément en émission et en réception. Parmi ces treize (13) vitesses, huit (8) sont obtenues simplement à partir du Registre de Sélection d'Horloge (CSR). Les cinq autres nécessitent en plus de CSR la programmation du bit sept (7) du Registre de Contrôle Auxiliaire.

Les sélecteurs d'horloge permettent la sélection indépendante de l'une de ces vitesses par l'émetteur ou le récepteur.

Quatre compteurs/horloges sont disponibles, un pour chaque bloc.

En mode horloge, le compteur/horloge génère une onde carrée dont la période est le double de celle de l'horloge chargée dans les deux registres associés au compteur/horloge (CTUR et CTLR.).

En mode compteur, le compteur/horloge décrémente le nombre de pulsations chargé dans les registres CTUR et CTLR jusqu'à épuisement. Une interruption (si programmée) est alors générée.

3.4.- Emission

L'octal-UART dispose de huit émetteurs asynchrones. Chaque émetteur sérialise les données reçues du processeur sous forme parallèle. Il rajoute automatiquement à la donnée émise, un bit START, une parité optionnelle et le nombre de bits STOP programmé. Si le registre d'émission est vide, le bit Emission Vide du Registre d'Etat est mis à 1. Il est remis à zéro, après chargement du registre.

3.5.- Réception

L'OCTAL-UART dispose de huit récepteurs asynchrones. Le récepteur transforme la donnée sérielle en parallèle. Le registre de réception consiste en une FIFO (First-In, First-Out), de trois positions mémoire. La donnée est chargée du registre à décalage dans la FIFO avec les trois bits d'état: Erreur de Parité, Erreur sur le Bit STOP, et Réception d'un BREAK. Le bit Réception Prête du Registre d'Etat SR est mis à 1 si un caractère au moins est disponible dans la FIFO, et le bit FIFO pleine est mis à 1 si les trois positions de la FIFO sont occupées (FIFO Pleine). Une lecture de la FIFO remet le bit FIFO Pleine à 0, alors que la remise à 0 du bit Réception Prête n'est effectuée que lorsque la FIFO est vide.

3.6.- Broches d'entrée à service multiple

L'octal-UART dispose de quatre ports d'entrée, chacun constitué de 8 broches (une broche par canal). L'état de ces broches peut être lu dans le Registre Port d'Entrée associé à deux canaux seulement.

3.7. - Caractéristiques générales

Les caractéristiques de ce composant sont les suivantes:

- 8 récepteurs/émetteurs asynchrones, full-duplex.
- Registres de réception à 3 positions mémoire + registres à décalage.
- Format de données programmable:
  - 5 à 8 bits de données plus parité,
  - parité paire, impaire, forcée, pas de parité,
  - longueur du bit STOP variant entre 9/16 et 1, 19/19 et 2 avec un incrément de 1/16-bit.
- Vitesses indépendantes en réception et en émission.
- Vitesses pour le récepteur et l'émetteur sélectionnées parmi:
  - 18 vitesses fixes entre 50 et 38.4K bauds,
  - 4 sources de vitesses dérivées du compteur/horloge associé à chacun des 4 blocs,
  - 1x ou 16x l'horloge externe.
- Contrôle d'erreur de parité, de débordement et de framing (non réception du bit STOP).
- Détection d'un bit START erroné.
- Génération et détection d'une coupure de ligne.
- Canal programmable en modes:
  - normal (full-duplex),
  - écho automatique,
  - bouclage en réception,
  - bouclage en émission.
- 4 sorties d'interruption avec chacune 8 sources d'interruptions masquables.

4.- INTERFACE DE COMMUNICATION SYNCHRONE/ASYNCHRONE DUAL [10]

La communication du côté de la voie composite est assurée par un interface de communication synchrone/asynchrone dual. Cet

interface de communication est aussi appelé DUSART (Dual Universal Synchronous-Asynchronous Receiver Transmitter). C'est le TS68564 de Thomson, présenté sous forme d'un boîtier de 48 broches.

Son apparition sur le marché en 1986 simplifia de manière considérable la communication sur la voie composite. En effet, jusqu'à cette date là, les interfaces de communication compatibles avec le processeur utilisé (6809 de Motorola), n'étaient pas dotés de protocole de communication. La gestion de la communication sur la voie haute vitesse incombait donc, au concepteur de l'application. Elle était entièrement logicielle.

Il comprend deux voies synchrones/asynchrones. L'une est utilisée en mode synchrone pour la voie composite, et l'autre en mode asynchrone pour la supervision.

L'interfaçage avec le processeur se fait à l'aide de:

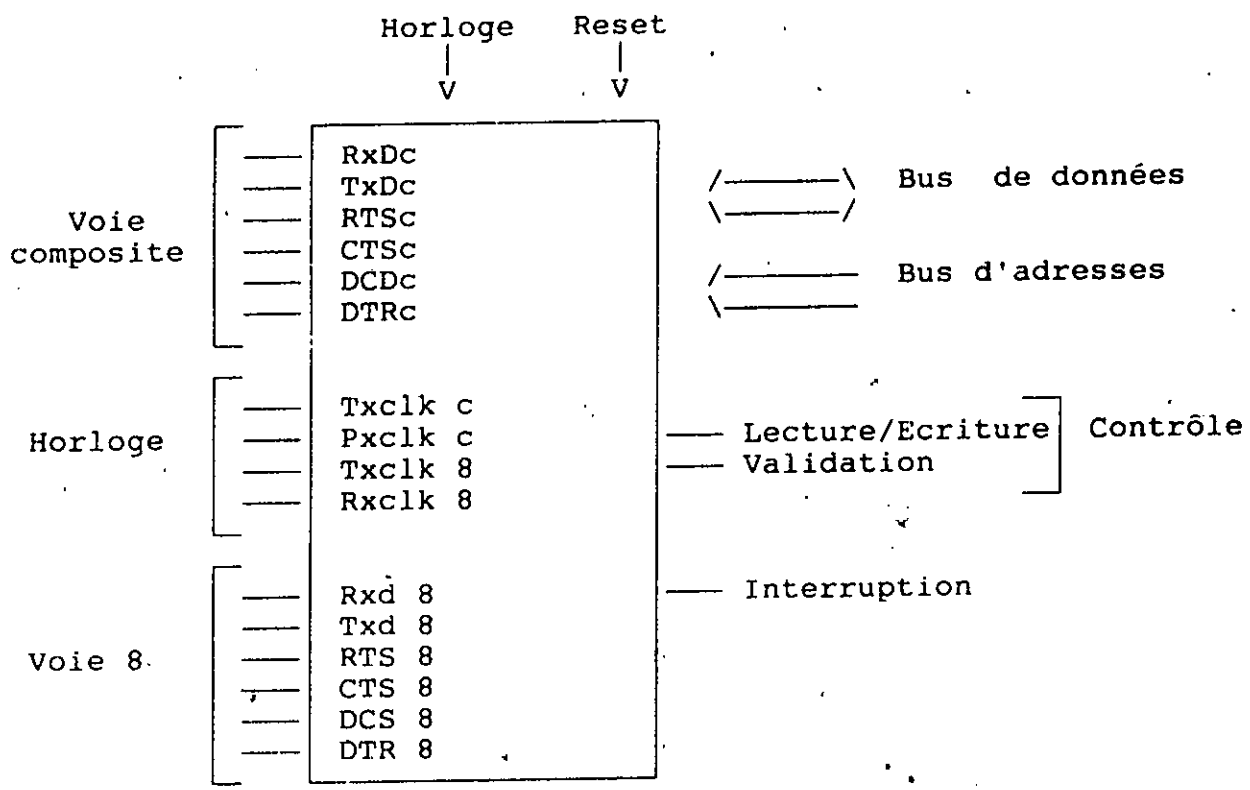


Figure 3.7 Interface voie composite.

— un bus de données bidirectionnel de 8 bits,
— un bus d'adresses de 5 lignes,
— deux lignes de contrôle,

— une ligne d'interruption,
— six lignes pour le contrôle du dialogue entre l'interface et le périphérique, par voie,
— une entrée d'horloge.

Le TS68564 comprend donc deux canaux indépendants, à double sens (full duplex). Chaque canal contient un émetteur, un récepteur, une logique de contrôle d'interruption, un générateur de vitesse du baud, dix registres à lecture/écriture et deux registres à lecture seulement.

Le récepteur est constitué d'une FIFO de trois positions mémoire et d'un Registre à Décalage.

Le DUSART offre le choix entre la boucle de surveillance (polling) et le mode d'interruption pour le transfert des données et le contrôle des états.

4.1.- Boucle de surveillance

En mode boucle de surveillance, toutes les interruptions sont désactivées. Les Registres d'Etat 0 et 1 sont remis à jour au moment approprié après chaque événement spécifique. Le processeur doit tester en permanence ces deux registres afin de s'enquérir des événements survenus.

4.2.- Interruptions

Le DUSART offre un schéma d'interruption bien élaboré (structure arborescente, voir figure 3.8), fournissant aux applications en temps réel, un temps de réponse aux interruptions, très rapide.

L'Interruption d'Emission, l'Interruption de Réception, l'Interruption de Réception Spéciale, et l'Interruption Externe sont les principales causes de l'interruption NMI. Elles sont différenciées à l'aide du Registre de Vecteur d'Interruption. Le fonctionnement de ce registre est validé par le paramètre Vecteur d'Interruption Affecté, dans le Registre de Contrôle d'Interruptions.

L'ordre de priorité de ces causes d'interruption est établi comme suit: Interruption de Réception, Interruption d'Emission, et Interruption Externe. Le canal A est plus prioritaire que le canal B.

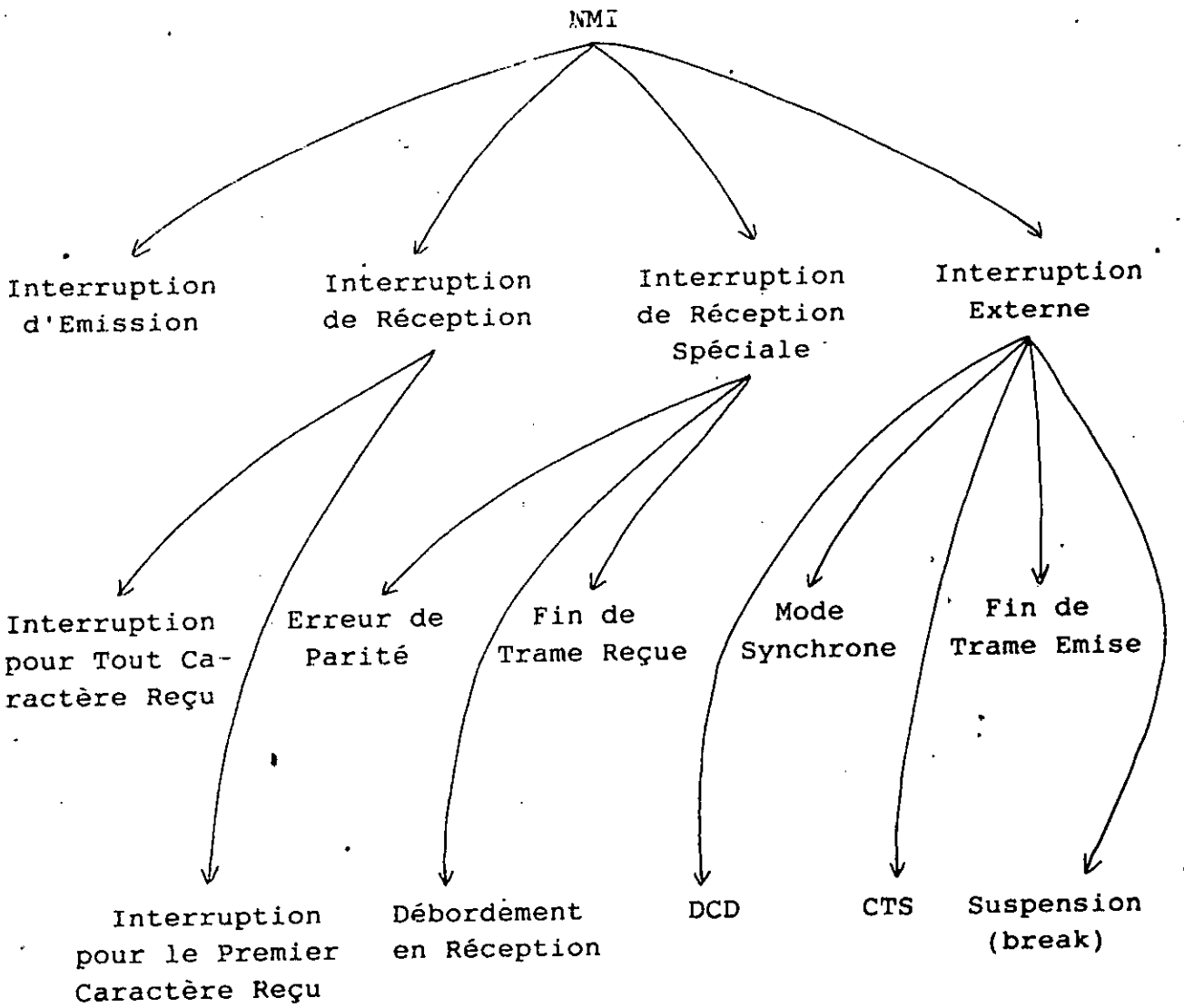


Figure 3.8. Structure des interruptions

Pour distinguer entre les trois sous-causes de l'Interruption de Réception Spéciale, il suffit de se référer au Registre d'Etat 1. Quant à la distinction entre les sous-causes de l'Interruption Externe, elle est réalisée à l'aide du Registre d'Etat 0.

4.2.1.- Interruption d'Emission

Lorsque l'Interruption d'Emission est validée, le processeur est interrompu à chaque vidage du Registre d'Emission. Cela suppose donc, que le Registre d'Emission doit être chargé afin de pouvoir se vider, et générer, par conséquent, une interruption.

4.2.2.- Interruption de Réception

Nous distinguons deux modes d'Interruption de Réception: l'Interruption pour le Premier Caractère Reçu et l'Interruption pour Tout Caractère Reçu.

L'Interruption pour le Premier Caractère Reçu est généralement utilisée pour démarrer une boucle de surveillance (polling). Une interruption est générée pour le premier caractère reçu, ensuite les seules interruptions qui surviennent, sont celles qui sont causées par une Réception Spéciale.

L'Interruption pour Tout Caractère Reçu est générée à la réception de chaque nouveau caractère ou Condition Spéciale.

4.2.3.- Interruption de Réception Spéciale

Une donnée arrive dans la FIFO de réception accompagnée de son état qui, quant à lui, est reçu dans la FIFO de réception d'erreur. Cet état peut être l'un des trois suivants: Débordement en Réception, Erreur de Parité ou Fin de Trame Reçue. On dit alors que l'on est dans une situation de Réception Spéciale. Cette situation est traduite par une Interruption de Réception Spéciale qui est automatiquement validée avec l'Interruption de Réception.

Lorsqu'une Erreur de Parité ou un Débordement en Réception surviennent, ils sont maintenus dans le Registre d'Etat 1, qui les signale, et le restent générant ainsi une interruption pour chaque nouveau caractère reçu, jusqu'à ce que la commande Réactiver Interruption de Réception Spéciale du Registre de Commande soit appelée.

La condition Fin de trame Reçue n'est pas maintenue dans le Registre d'Etat 1. Elle disparaît au prochain caractère reçu.

4.2.4.- Interruption externe

La principale fonction d'une Interruption Externe est de signaler les signaux de transition des broches CTS, DCD et SYNC. Elle peut cependant, être également causée par une Fin de Trame Emise ou une suspension de l'exécution du programme (Break/Abort).

Lorsque l'une de ces cinq causes d'interruption arrive, elle est maintenue dans le Registre d'Etat 0, et ne peut être satisfaite qu'à l'aide de la commande Réactiver Interruption Externe.

5.- PORT D'ENTREE/SORTIE

Le microprocesseur a besoin de connaître et de contrôler pendant son traitement, l'état des canaux de transmission à tout instant. Ces états sont fournis par les ports d'entrée/sortie intégrés aux interfaces de communication, à l'exception de l'état DCD. Pour cela un port d'entrée/sortie séparé est utilisé pour générer les signaux DCD en sortie, pour les 8 voies basse vitesse. Ces signaux sont lus à travers le bus de données.

6.- PARTITIONNEMENT DE LA MEMOIRE

Le transfert des adresses du microprocesseur vers le bus d'adresses du système se fait par l'intermédiaire de seize (16) lignes unidirectionnelles. Ces lignes permettent d'adresser:
— les différentes mémoires du multiplexeur, à savoir: la mémoire programmable (EPROM), la mémoire tampon (RAM), et la mémoire non volatile (NVRAM),
— les deux interfaces de communication,
— le port d'entrée/sortie,

A cet effet, l'espace adressable (64 K octets) a été subdivisé en six (6) zones de tailles différentes. Le partitionnement de cet espace est décrit dans le tableau ci-dessous.

La mémoire tampon emmagasine l'information échangée. La mémoire non volatile sauvegarde les caractéristiques des différents périphériques (vitesse d'émission, vitesse de réception, parité...). Quant à la mémoire programmable, elle comprend le programme exécutable, régissant le multiplexeur.

Adresse	Espace	Taille
0000 H 1FFF H	Non utilisé	-
2000 H ---- H	Mémoire - tampon RAM	-
---- H 3FFF H	Non utilisé	-
4000 H 4000 H	Port d'E/S	1 Octet
4001 H 4110 H	Non utilisé	-
4111 H 44FF H	Mémoire non volatile NVRAM	1 K
4500 H 4FFF H	Non utilisé	-
5000 H 501F H	Interface de communication DUSART	31 Octets
5020 H 5FFF H	Non utilisé	-
6000 H 603F H	Interface de communication OCTAL-UART	63 Octets
6040 H 7FFF H	Non utilisé	-
8000 H 9FFF H	Mémoire programmable EPROM	8 K
A000 H FFFF H	Non utilisé	-

7.- EMETTEURS/RECEPTEURS DE LIGNE

Sachant que les différents circuits constituant la carte fonctionnent avec des niveaux de tension de 0 ou 5 volts, et qu'au niveau des jonctions RS232 constituant l'interface entre périphériques et multiplexeur, on trouve des tensions de -12 et +12 volts, des émetteurs/récepteurs de lignes sont utilisés pour convertir le +5V au -12V et le 0V au +12V.

8.- JONCTIONS RS232

L'interface électrique du multiplexeur avec les périphériques et le modem comprend des jonctions EIA RS232 ou CCITT V24/V28, femelles, 25 broches. Parmi les principaux signaux échangés, nous retrouvons:

— RTS (Request To send): demande d'émission du périphérique;
— CTS (Clear To send): réponse du multiplexeur au RTS; multiplexeur prêt à recevoir;
— DTR (Data Terminal Ready): périphérique prêt à émettre et/ou à recevoir;
— DCD (Data Carrier Detect): détection de porteuse;
— Txd (Transmit Data): émission des données;
— Rxd (Receive Data): réception des données.

CHAPITRE 4

PRODUIT SIMULE

1. - QU'EST-CE-QUE LA SIMULATION ?

1.1.- INTRODUCTION

Afin de répondre aux exigences continuellement croissantes de la technique moderne, la nécessité de recourir à des techniques de résolution plus fines, se fait de plus en plus pressante.

Ainsi, à mi-chemin, entre les méthodes analytiques exactes, et les méthodes heuristiques approximatives, la simulation occupe une place de choix dans la conception et le développement des systèmes physiques et économiques. C'est aujourd'hui, un procédé d'usage général.

En effet, à l'heure actuelle, on parle beaucoup de simulation: simulation dans le domaine financier, industriel et technique, biologique ou social. Comment peut-on alors définir la simulation?

1.1.1.- Définition de la simulation [16]

La simulation est en fait une méthode d'étude de processus ou système, consistant à remplacer celui-ci, par un modèle plus simple ayant un comportement semblable ou analogue. Dans le cas de processus industriels, la simulation préalable permet d'envisager toutes les situations normales ou accidentelles. Elle doit permettre aussi à l'extrême, de connaître les conséquences de l'imprévisible. Elle constitue une assurance vis-à-vis des erreurs de conception. Elle intervient beaucoup dans la recherche et l'amélioration des techniques.

Ainsi, réaliser une simulation, c'est construire un système analogue au système réel, dans un environnement différent, pour une étude plus aisée.

La simulation est en général appliquée:

— comme procédé explicatif qui aiderait à cerner, définir, puis formuler un problème donné,
— comme outil d'analyse extrêmement poussée d'un quelconque phénomène, analyse de son évolution au cours du temps, de ses caractéristiques, voire de ses capacités de réaction à un effet donné,
— comme outil d'évaluation, de comparaison et finalement de prise de décision au vu d'un certain nombre de solutions au problème posé,
— comme outil de prévision, de planification à court, moyen et long terme.

1.1.2.- Intérêt de la simulation [16]

Les principaux intérêts de la simulation se résument ainsi:

— gain financier non négligeable, dans l'étude, la mise au point, le développement et la vente d'un produit, pour des systèmes complexes, modélisables, étudiés dans les plus diverses configurations proches de la réalité (réacteur nucléaire, missile, engin guidé),
— possibilité d'étudier le système soit en temps réel où le modèle réagit à la même vitesse que le système réel, soit en temps accéléré ou ralenti. Dans ce cas, le modèle réagit respectivément plus ou moins vite que le système réel.

Si le temps ralenti permet d'étudier ou d'analyser un phénomène plus finement, le temps accéléré permet, dans un délai très court (temps de calcul, durée de la simulation), de prévoir les réactions, les conséquences à une action provoquée sur le système simulé dans un espace de temps équivalent très long.

Le temps réel est très intéressant: il permet des comparaisons et même des rebouclages de résultats entre le modèle de simulation et le système réel à étudier. Grâce à une simulation fine, ce modèle, très proche de la réalité est unique. La mise au point du système réel est alors très rapide. De même, une partie de ce dernier peut être introduite dans la simulation elle même (exemple: introduction du bloc réel de pilotage/guidage dans la simulation d'un engin guidé).

1.2.- SYSTEMES ET MODELES

Le système et le modèle sont les deux notions fondamentales de la simulation. Il serait donc peu plausible de parler de simulation sans aborder ces deux concepts de base.

1.2.1.- Le système

Le système est un ensemble d'objets (appelés éléments ou entités) en interaction, réunis pour remplir une fonction bien déterminée.

Tout système évolue dans un environnement qui constitue son milieu extérieur. Selon que le système soit ouvert ou fermé, il est en relation ou non avec cet environnement. Dans le cas affirmatif, son comportement est affecté par les perturbations extérieures, souvent considérées comme des entrées du système.

Un système est caractérisé par son état qui est en perpétuel changement. Cet état à un instant donné, n'est rien d'autre que l'ensemble des valeurs attribuées aux entités du système (ou valeurs des attributs).

De ce fait, la simulation va se proposer de:

— décrire le système à chaque instant: portrait statique du système,
— représenter les changements d'état du système au cours du temps: portrait dynamique du système.

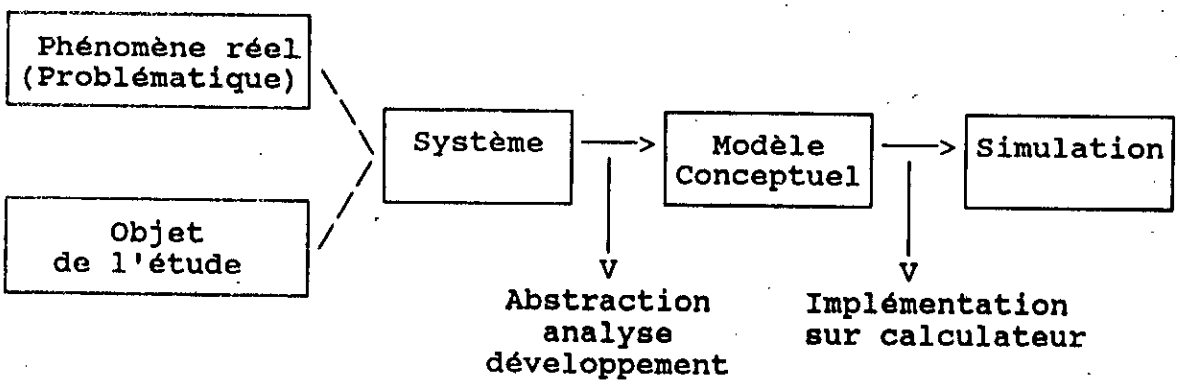
Il reste donc à comprendre comment va procéder la simulation pour décrire ces deux aspects du système. C'est ainsi que nous aboutissons au deuxième concept de base de la simulation: le modèle.

1.2.2.- Le modèle

Le modèle est une représentation abstraite qui extrait l'essence du système. Or, le principe fondamental de la simulation est que «tout phénomène réel, quel qu'il soit, peut être représenté par un système». Partant de ce principe, nous pouvons extrapoler

la définition du modèle en affirmant que ce dernier est l'abstraction du phénomène réel à étudier.

Ainsi, les liens qui relient le phénomène réel, objet de l'étude, à la simulation, peuvent être représentés comme suit:



1.3.- TYPES DE SIMULATION

On distingue trois sortes de simulation selon le type de modèles qu'elles permettent de construire: la simulation continue, la simulation discrète et la simulation combinée.

1.3.1.- Simulation continue [17]

En simulation continue les variables d'état changent de valeur instantanément au cours du temps: les variables dépendantes sont des fonctions continues de la variable dépendante, qui peut être, par exemple, le temps; on dit que le système évolue continûment.

Un modèle continu est décrit soit par des équations différentielles ordinaires sur la variable du temps, soit par des équations aux dérivées partielles qu'il s'agit d'intégrer pour connaître le comportement du modèle suivant les conditions initiales, les valeurs numériques de ses paramètres, etc.

A titre d'exemple de systèmes étudiés de cette façon, on peut citer:
— le comportement d'une structure aérodynamique,
— des processus biologiques, chimiques et physiques,
— des problèmes d'écologie, de pollution et d'environnement.

1.3.2.- Simulation discrète [17]

En simulation discrète, les changements d'état du modèle sont discrets: les valeurs des variables dépendantes changent de façon discrète au cours du temps; on dit que le système évolue par sauts.

Comme exemples de systèmes décrits par des modèles de simulation discrète, on peut citer:

— le trafic, la communication,
— la mise en oeuvre d'un système informatique,
— la gestion des stocks, de la production,
— l'entretien d'un parc de machines.

Trois concepts de base interviennent pour traduire les modèles de simulation discrète: l'activité, l'événement et le processus.

Pendant tout intervalle de temps où l'état d'une entité du système ne change pas (ses attributs ne changent pas de valeur), on dit que cette entité est engagée dans une activité.

Dès qu'un événement a lieu, l'état de l'entité change. Ainsi, nous pouvons affirmer qu'au cours du temps, l'entité s'engage dans une suite d'activités, le passage d'un état à un autre étant occasionné par l'occurrence d'événements.

La succession des activités dans lesquelles s'engage l'entité est appelée processus.

En s'appuyant sur ces concepts et sur la dualité entre l'activité et l'événement, nous obtenons le schéma suivant:

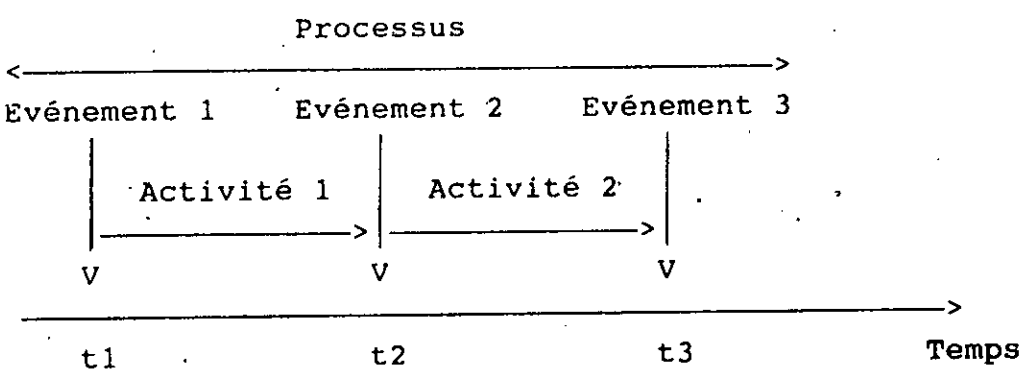


Figure 4.1. Relation entre activité, événement et processus

Ainsi, un modèle de la simulation discrète peut être réalisé selon trois approches: par événement, par activité ou par processus.

— **L'approche par événements**

Cette approche consiste à élaborer le modèle en une succession d'événements et décrire ainsi les changements d'état provoqués par chaque événement. La méthode utilisée pour avancer le temps sera donc celle «du prochain événement».

— **L'approche par activités**

Cette approche est la duale de la précédente. On analyse le système non plus pour répertorier les événements qui provoquent le changement d'état, mais les intervalles de temps où le système est stable. Les conditions d'altération de la stabilité de ce système, (les événements) sont alors à définir. Le mécanisme utilisé pour avancer le temps, dans cette approche est une horloge.

— **L'approche par processus**

C'est la méthode conceptuelle, la plus utilisée. Elle combine les deux approches évoquées ci-dessus. On obtient ainsi une succession d'activités induites par des événements.

1.3.3.- Simulation combinée [17]

Dans un modèle de simulation combinée, certaines variables décrivant le système réel sont discrètes, d'autres continues, d'autres encore continues par intervalle. On distingue alors deux types d'événements:

— les événements temporels qui ont lieu à des instants fixes, prédéterminés;
— les événements d'état qui sont déclenchés lorsque le système atteint un état défini par certaines conditions, par exemple lorsqu'un certain seuil est dépassé.

Il y a trois types principaux d'interactions entre les variables continues et discrètes dans les modèles combinés:

— une variable continue peut subir un changement discret.
Exemple: la connexion d'une nouvelle centrale nucléaire au réseau électrique accroît l'énergie électrique disponible d'une quantité déterminée.
— Lorsqu'une variable continue atteint une valeur limite, elle déclenche un événement.
Exemple: si dans un tunnel routier la quantité de monoxyde de carbone dépasse une certaine limite, il y a fermeture du tunnel et arrêt du trafic.
— La loi de comportement dynamique d'une variable continue peut être changée à un moment précis.
Exemple: la loi de croissance d'une espèce biologique peut être différente après une pollution chimique causée par un accident.

Ce n'est que depuis le début des années soixante-dix, que l'utilité de représenter certains systèmes réels à l'aide de modèles combinés est apparue clairement et que des logiciels, de plus en plus nombreux, sont offerts.

1.4.- LANGAGES DE SIMULATION [17, 21]

De nombreux logiciels sont actuellement disponibles dans le domaine de la simulation sur ordinateur numérique. les principes sur lesquels ils sont fondés, les éléments de base qu'ils contiennent et le cadre conceptuel qu'ils fournissent à l'utilisateur pour la construction du modèle sont souvent très spécifiques.

En principe, le choix d'un langage se fait en fonction des qualités qu'on lui reconnaît, telles que:
— flexibilité,
— cadre conceptuel bien adapté aux problèmes à résoudre,
— syntaxe et sémantique faciles à mémoriser,
— possibilité de décomposer un système complexe en sous-systèmes beaucoup plus simples, permettant une construction évolutive et modulaire des modèles;
— dispositifs offerts pour la collecte de statistiques et la sortie des résultats;
— moyens disponibles pour la mise au point des programmes et la détection des erreurs.

En pratique, la diffusion du langage et sa disponibilité sur l'ordinateur utilisé sont souvent décisives.

1.4.1.- Langages de simulation continue

Les logiciels de simulation continue disponibles actuellement permettent essentiellement d'étudier le comportement de systèmes décrits à l'aide d'équations différentielles ordinaires. Ils possèdent notamment une bibliothèque de routines d'intégration pour faciliter l'écriture des programmes de résolution de ces systèmes.

Un ordinateur numérique est par nature discret. Si on l'emploie pour la simulation continue, le logiciel utilisé devra résoudre deux types de problèmes: l'intégration et la simultanéité.

— Les routines d'intégration représentent le coeur de tout logiciel de simulation continue. De nombreux algorithmes doivent être disponibles pour la résolution de différents types de problèmes. De plus, l'utilisateur doit pouvoir en incorporer facilement d'autres si nécessaire.
— Les processus réels que l'on veut simuler se déroulent très souvent simultanément, or les ordinateurs numériques actuels, exécutent les instructions séquentiellement, dans un ordre défini. Le logiciel de simulation continue doit donc offrir à l'utilisateur la possibilité de définir des sections procédurales et des sections structurales. Dans le premier cas, il décrit et contrôle lui-même la séquence d'exécution et dans le second, il laisse à un programme de tri le soin de le faire.

CSMP III et DYNAMO sont deux exemples de langages de simulation continue (voir annexe A).

1.4.2.- Langages de simulation discrète

Le problème fondamental à résoudre en simulation discrète est celui de la détermination dynamique de la séquence des événements à réaliser.

L'ordonnancement des événements est effectué à l'aide d'un échéancier. Ce dernier contient une liste des événements ordonnés par date d'occurrence croissantes; à dates égales, des règles d'ordonnancement peuvent être précisées, sinon la règle FIFO (First In, First Out) est appliquée. Un programme de gestion de l'échéancier parcourt celui-ci et fait exécuter les changements d'état correspondant aux événements rencontrés. Une horloge est tenue à

jour en permanence. Le temps courant simulé est fixé par la date de l'événement en cours de réalisation.

Dans l'approche par activités, le logiciel comprend une routine de gestion dynamique d'horloges associées à certaines entités du modèle. La valeur minimale référée à chaque étape par ces horloges devient le temps courant simulé. Après chaque activité, les conditions de son déclenchement sont définies; elles sont évaluées chaque fois que le temps courant de simulation est changé. Une activité n'est déclenchée que si ses conditions sont satisfaites, et l'état du système est alors modifié en conséquence.

Actuellement, seuls les logiciels basés sur l'approche par événements (SIMSCRIPT et GASP) et selon l'approche par processus (GPSS et SIMULA) sont couramment utilisés (voir annexe A).

1.4.3.- Langages de simulation combinée

Les logiciels destinés à la programmation de modèles combinés discrets-continus, sont pour la plupart d'entre eux, des extensions de langages de simulation discrète. Ils comportent trois parties principales:

— une partie discrète dont les éléments sont empruntés au logiciel de simulation discrète,
— une partie continue permettant la réalisation de modèles de simulation continue,
— un programme de contrôle comprenant les algorithmes décrivant le passage de la simulation discrète à la simulation continue et inversement.

Le passage de la simulation discrète à la simulation continue est effectué selon le principe suivant: l'horloge est avancée au temps t1 indiqué par l'événement à réaliser; le programme correspondant à cet événement est exécuté, entraînant les changements d'état du système qu'il décrit. Avant d'avancer l'horloge au temps t2 de l'événement futur suivant, le programme de contrôle inspecte le modèle pour chercher s'il y a des équations différentielles à résoudre pour l'intervalle de temps considéré; si c'est le cas, il donne le contrôle aux programmes de simulation continue. Le passage inverse est un peu plus délicat. Les programmes de simulation continue sont exécutés soit jusqu'au temps t3 où un événement

temporel doit avoir lieu, soit jusqu'à ce que les conditions soient réalisées pour qu'un événement d'état soit déclenché. Initialement c'est la partie simulation discrète qui reçoit le contrôle.

Comme exemples de langages de simulation combinée, on peut citer GASP IV et SLAM (voir annexe A).

1.5.- SLAM II [20]

Notre choix du langage SLAM II fut principalement motivé par sa disponibilité dans notre milieu de travail. De surcroît, il présente de grandes qualités de flexibilité et répond parfaitement à notre problématique, à savoir, la construction d'un réseau de communication.

Ce paragraphe sera consacré à une présentation générale du langage, ses concepts de base et ses diverses possibilités. Des compléments d'information concernant notamment les outils de modélisation offerts par SLAM et les résultats obtenus en sortie seront donnés au cours de notre application.

1.5.1.- Présentation générale du langage

SLAM sont les initiales de «Simulation Language of Alternative Modeling». La première version du langage (SLAM I) a été élaborée en 1979 puis améliorée en 1984 (SLAM II).

Le logiciel est développé par A.B. Pritsker. Il comporte une version exploitable sur mini-ordinateur et une autre exploitable sur micro-ordinateur. Cette dernière est actuellement disponible en deux versions:

— une version standard mise sur le marché;
— une version éducative, réplique de la précédente mais qui ne peut être utilisée pour les systèmes de plus grande taille.

Une présentation détaillée de SLAM II se trouve dans l'ouvrage «Introduction to Simulation and SLAM II» écrit par A.B. Pritsker [20].

1.5.2.- Possibilités offertes par SLAM II

SLAM II est un langage de simulation combinée siscrète-continue, écrit en Fortran. Il permet de décrire la partie discrète du modèle soit selon l'approche par processus, soit selon l'approche par événements ou encoré en combinant les deux.

L'approche par processus consiste à construire à l'aide des blocs disponibles appelés noeuds (en tout 24), un réseau dans lequel se déplaceront des entités. Ces entités sont caractérisées par un vecteur qui conserve leurs attributs. Les noeuds permettent de créer, détruire, diriger et traiter des entités, gérer des files d'attente, etc...

S'il utilise l'approche par événement, le modéliste décrit à l'aide de sous-programmes, écrits en Fortran, les changements d'état du système causés par ces événements.

La partie continue du modèle est définie soit par des équations différentielles, soit par des équations aux différences qui décrivent le comportement dynamique des variables d'état (ou attributs continus). SLAM II met à la disposition de l'utilisateur des tableaux spéciaux pour stocker les valeurs des variables d'état ainsi que leurs dérivées (SS, DD). Le calcul des valeurs des variables d'état est réalisé à l'aide d'algorithmes d'intégration avec incrémentation du temps par pas adaptables.

La construction d'un modèle combiné discret-continu est facilitée par le fait que tous les différents éléments peuvent interagir:

— les entités se déplaçant dans le réseau peuvent déclencher des événements discrets,
— les événements peuvent modifier le déplacement des entités dans le réseau,
— les entités peuvent causer des changements instantanés des valeurs des variables d'état,
— lorsqu'elles franchissent un seuil prédéterminé, les variables d'état peuvent créer des entités dans le réseau ou encore des événements,
— des événements peuvent provoquer des changements instantanés des valeurs de certaines variables d'état,
— certaines valeurs de variables d'état peuvent provoquer l'occurrence d'événements.

Bien qu'il permette de construire rapidement des modèles simples, de les exploiter, puis de les compléter au fur et à mesure des besoins, SLAM souffre du handicap de ne pas permettre la programmation structurée.

1.5.3.- La structure du programme SLAM II

Le programme SLAM est composé de deux grandes parties:
— le programme principal (Main program),
— les programmes périphériques.

1.5.3.1.- Le programme principal

Ce programme va essentiellement organiser l'espace mémoire disponible (matrice NSET/QSET de dimension 16000 mots pour la version standard), répartir cet espace entre les différentes composantes du modèle telles que les files, les attributs et les résultats statistiques de la simulation (tableaux, rapports de sortie, etc...). Le logiciel contient déjà le programme principal, et la matrice NSET/QSET est prédimensionnée. Aussi, le concepteur n'aura pour tâche que de transcrire les programmes périphériques.

1.5.3.2.- Les programmes périphériques

Les programmes peuvent être:
— un réseau SLAM,
— des sous-programmes Fortran,
— une combinaison des deux.

Le réseau SLAM est transcrit en instructions SLAM, chaque instruction correspond à un noeud ou une branche du réseau.

Les sous-programmes Fortran peuvent faire appel à des fonctions SLAM prédéfinies dans la librairie. L'appel se fait à l'aide de l'instruction CALL. Ces sous-programmes qui correspondent à des événements peuvent être introduits dans le réseau SLAM à l'aide des deux instructions SLAM, EVENT et ENTER.

La jonction entre programme principal, sous-programmes et réseau SLAM se fait grâce aux librairies SLAM et FORTRAN.

1.5.4.- Les instructions de contrôle de la simulation

Grâce à un certain nombre d'instructions de contrôle, nous parvenons à établir les conditions initiales de la simulation, commander le temps de début de collecte des observations, l'impression ou non des résultats intermédiaires et des messages d'erreur. Nous disposons également de nombreux moyens de détection d'erreurs et de mise au point du programme.

1.6.- METHODOLOGIE DE LA SIMULATION

Le cadre général méthodologique de la simulation comporte dix étapes essentielles qui constituent la démarche de travail de toute personne désirant résoudre un problème par simulation. Ces étapes sont consignées sur l'organigramme de la figure suivante.

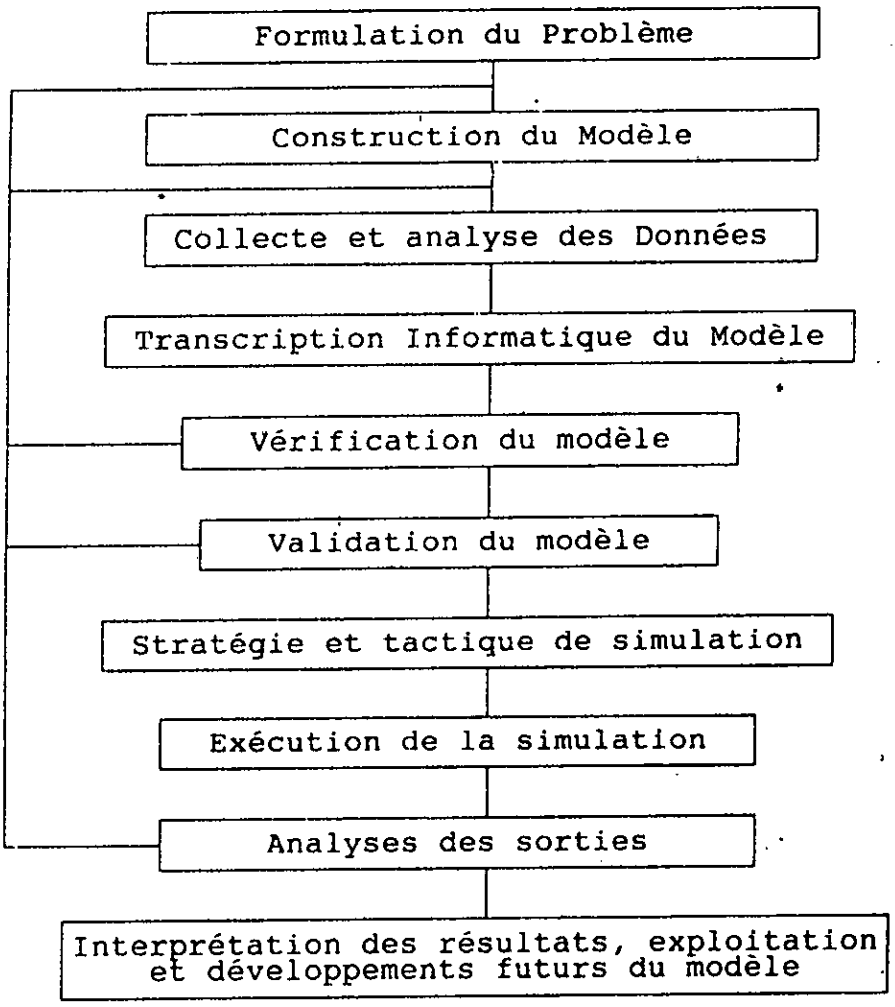


Figure 4.2. Méthodologie de simulation

1.6.1.- Formulation du problème

Le concepteur doit cerner, définir, identifier en toute clarté le problème, et cela sans négliger aucun aspect. Il s'agira de définir les frontières du système, ses éléments et leurs caractéristiques, et la composition de ses entrées et sorties. Il s'avère souvent que ce processus de formulation du problème n'est jamais totalement achevé, car au cours de la simulation, des facteurs nouveaux, des interrogations ou préoccupations jusque là insoupçonnées, viennent se greffer au problème de départ, complétant et précisant ainsi les objectifs initiaux.

1.6.2.- Modélisation

Comme nous l'avons précédemment évoqué, le modèle constitue le lien entre le phénomène réel, objet de l'étude et la technique d'étude: la simulation. Il est par conséquent facile de saisir l'importance que revêt en simulation la construction du modèle ou modélisation.

Pour ce faire, le concepteur devra faire appel à toute sa perspicacité pour établir les deux paramètres fondamentaux qui prédéterminent une bonne modélisation, à savoir:

— une bonne définition du système, accompagnée de la meilleure identification de ses frontières,
— un choix judicieux du niveau de détail à inclure dans le modèle: savoir quels éléments du système inclure dans le modèle, lesquels négliger, quelles sont les interactions à prendre en compte, sur quelles variables baser la logique de changement d'état du système, le tout en tenant compte des objectifs de l'étude.

La définition du système doit se faire sous deux aspects:

— l'aspect statique qui consiste en l'identification des entités, leurs attributs, les entrées et les sorties.
— l'aspect dynamique où il s'agira de décrire les interactions entre les entités afin de pouvoir saisir et transcrire la logique de changement d'état du système.

Si ces deux aspects de la modélisation sont bien maîtrisés, nous pouvons nous déclarer optimistes quant à sa réussite.

1.6.3.- Collecte et analyse des données

Une fois les deux étapes précédentes abordées, il s'agira de passer à l'identification des données d'entrées que nécessite le modèle. Il faudra ensuite, acquérir ces données, et éventuellement les ajuster à des lois de probabilité connues en faisant appel aux tests d'inférence statistique. Cette phase d'identification, d'acquisition et d'ajustement des données d'entrée est primordiale en simulation car il arrive très souvent que le système à simuler soit extrêmement sensible à ces données. Aussi, des erreurs dans les données d'entrée peuvent conduire à des résultats n'ayant aucune commune mesure avec le système réel.

1.6.4.- Transcription informatique du modèle

Au cours de cette étape, le concepteur s'attellera à transcrire le modèle sous forme exploitable par l'ordinateur. Là encore, une alternative se présente pour le choix du langage à utiliser:

— soit programmer le modèle en utilisant un langage de programmation général du type FORTRAN ou PASCAL,
— soit utiliser un langage de simulation (SLAM, MAP/1, Q-GERT, GPSS...) et profiter des avantages qu'il offre pour faciliter l'élaboration du modèle (voir 1.4).

Concernant l'implantation du modèle sur ordinateur, le concepteur devra accorder une attention particulière, quant au choix de la machine, compte tenu, entre autres, des exigences du langage utilisé et de la taille du modèle élaboré.

1.6.5.- Vérification du modèle

Dans cette étape il s'agit essentiellement de vérifier si le modèle transcrit sur ordinateur exécute bien ce que le concepteur attend de lui. La tâche de l'utilisateur sera donc de déceler de quelconques aberrations dans cette transcription. Pour mener à bien cette mission, souvent laborieuse mais combien nécessaire, le concepteur dispose de plusieurs méthodes. Des méthodes manuelles de vérifications proposées par Kiviat et Fishman [23] permettent de suivre l'évolution dynamique du système sur un laps de temps assez court.

1.6.6.- Validation du modèle

Cette étape complète la précédente en ce sens qu'elle consiste à évaluer les performances du modèle, sur sa capacité à bien refléter le système qu'il est censé décrire. Ainsi, valider, c'est s'assurer de l'efficacité du modèle et des résultats qui découleront de la simulation de ce modèle. Ce n'est qu'à l'issue d'une validation réussie que l'on pourra affirmer que la décision prise expérimentalement au niveau du modèle aura le même impact dans la réalité. Certes, un modèle de simulation ne représente qu'une approximation du système. Il serait par conséquent irréaliste de prétendre atteindre une validité absolue du modèle. Afin d'approcher le plus possible cette validité absolue, le concepteur peut s'atteler à l'approche en trois phases développée par Naylor et Finger [26], présentée dans notre application.

1.6.7.- Stratégie et tactique de simulation

Il s'agit, au cours de cette étape, d'établir les conditions expérimentales d'exécution de la simulation.

1- Stratégie de simulation

Elaborer cette stratégie consiste à développer un plan d'expérience basé sur la combinaison optimale des valeurs des variables dites «de contrôle» de la simulation, à savoir, le nombre d'exécutions, la durée de chaque exécution et un certain nombre d'autres paramètres, le tout afin de maximiser l'efficacité de la simulation. L'élaboration de cette stratégie est particulièrement fastidieuse car chaque système possède sa spécificité et le concepteur doit souvent faire appel à son bon sens et procéder par tâtonnements, d'où un investissement important en temps.

2- Tactique de simulation

Cette tactique concerne trois aspects importants de la simulation:

— la détermination des conditions initiales de la simulation,
— la durée de la simulation,
— l'application des techniques de réduction de la variance des résultats obtenus à l'issue de la simulation.

1.6.8.- Exécution de la simulation

Compte tenu des étapes précédentes, il s'agira de «faire tourner le modèle» puis recueillir les résultats obtenus.

1.6.9.- Analyse des sorties

A l'issue des premiers résultats obtenus, le concepteur se doit de mieux saisir les mesures de performances du modèle en les analysant à travers les variations du courant aléatoire «stream» et de la longueur de la simulation. Pour ce faire, diverses procédures permettent la construction d'un intervalle de confiance pour chaque paramètre de mesure de performance.

1.6.10.- Interprétation des résultats, exploitation et développements futurs du modèle

C'est le stade de finalisation de l'étude. Une fois la fiabilité des résultats prouvée, le chercheur pourra passer à leur interprétation. Cette étape reste particulièrement délicate surtout lorsqu'il s'agit, par exemple, de mesurer, après élaboration de plusieurs scénarios du modèle, l'impact de plusieurs décisions pour sélectionner la meilleure. Enfin, une évaluation des perspectives d'exploitation du modèle élaboré pour de nouvelles préoccupations, peut être envisagée.

Remarque:

Comme nous avons pu le constater, ces étapes qui constituent le cadre méthodologique de la simulation sont loin d'être cloisonnées et indépendantes. Bien au contraire, les retours en arrière et les anticipations sont parfois d'une telle fréquence que la simulation en devient une technique de longue haleine demandant une très grande disponibilité. En effet, les faux départs, les interprétations erronées, les contraintes de différents ordres font que plusieurs formulations du problème sont souvent nécessaires, et de nombreuses modélisations sont tentées avant d'arriver à une base de travail solide et fiable.

2.- SIMULATION DU SYSTEME

L'objectif principal de notre étude par simulation est d'effectuer une analyse du système «multiplexeur» pour élaborer des scénarios qui tendent à optimiser sa capacité de fonctionnement. Il s'agira donc, d'analyser l'incidence de toute configuration sur les performances du système. Ces configurations reposeront sur:

— une variation de la longueur de la trame,
— une variation des vitesses,
— une variation dans les temps de service,
— une variation dans le taux de concentration.

Les paramètres d'optimisation seront:

— la taille des files d'attente à savoir, les mémoires tampons de réception et d'émission, et la file d'attente de trames du système,
— le temps d'attente dans ces files.

Par ailleurs, ce modèle pourra constituer le noyau d'un véritable réseau de multiplexeurs, travail pouvant faire l'objet d'une extension du projet actuel.

2.1.- DESCRIPTION DU SYSTEME

Sa principale fonction est la gestion des communications de deux équipements informatiques, sur deux sites, éloignés, reliés à l'aide d'une ligne de transmission à longue distance unique.

Sur chaque site, un équipement de huit (8) matériels informatiques (terminaux, ordinateurs, imprimantes, etc...), un terminal de supervision et un modem sont connectés au multiplexeur.

Le modem protège l'information transmise des erreurs et perturbations dues à l'état de la ligne de transmission. Le terminal superviseur assure la configuration des équipements informatiques connectés au multiplexeur et établit les liaisons de communication.

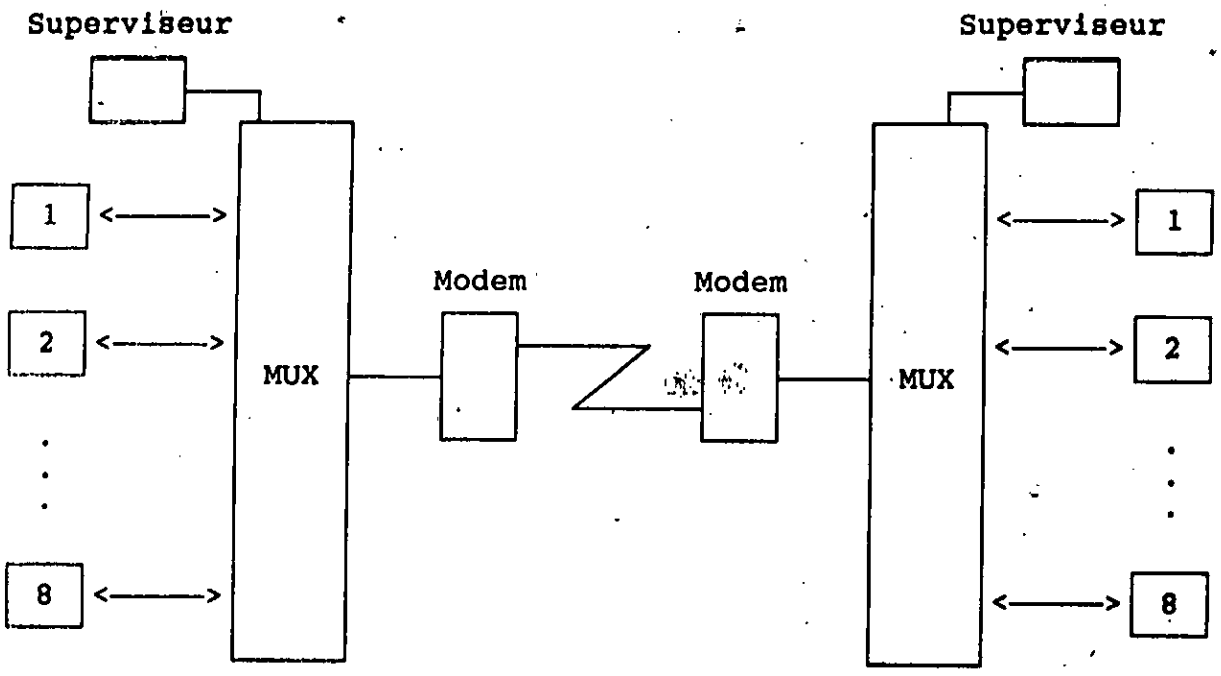


Figure 4.3. Communication par multiplexage

2.1.1.- Décomposition du système en processus

L'information échangée entre deux points de deux sites éloignés transite par trois circuits.

— Le premier circuit, appelé voie basse vitesse, la conduit de la station de travail locale à l'appareil de multiplexage, toujours local.
— Le second circuit, appelé voie haute vitesse, la conduit de l'appareil de multiplexage local à son homologue, situé à distance.
— Le troisième circuit, appelé voie basse vitesse, la conduit de l'appareil de multiplexage à distance à la station de travail destinataire.

Ce cheminement est effectué à l'aller et au retour pour les huit (8) liaisons point à point (voir figure 4.4).

A partir de cette simple définition surgissent d'ores et déjà dix-huit (18) processus, à savoir:
— huit processus de réception sur une voie basse vitesse qui récupèrent l'information reçue sur une voie basse vitesse pour la stocker dans la mémoire tampon de réception correspondante;

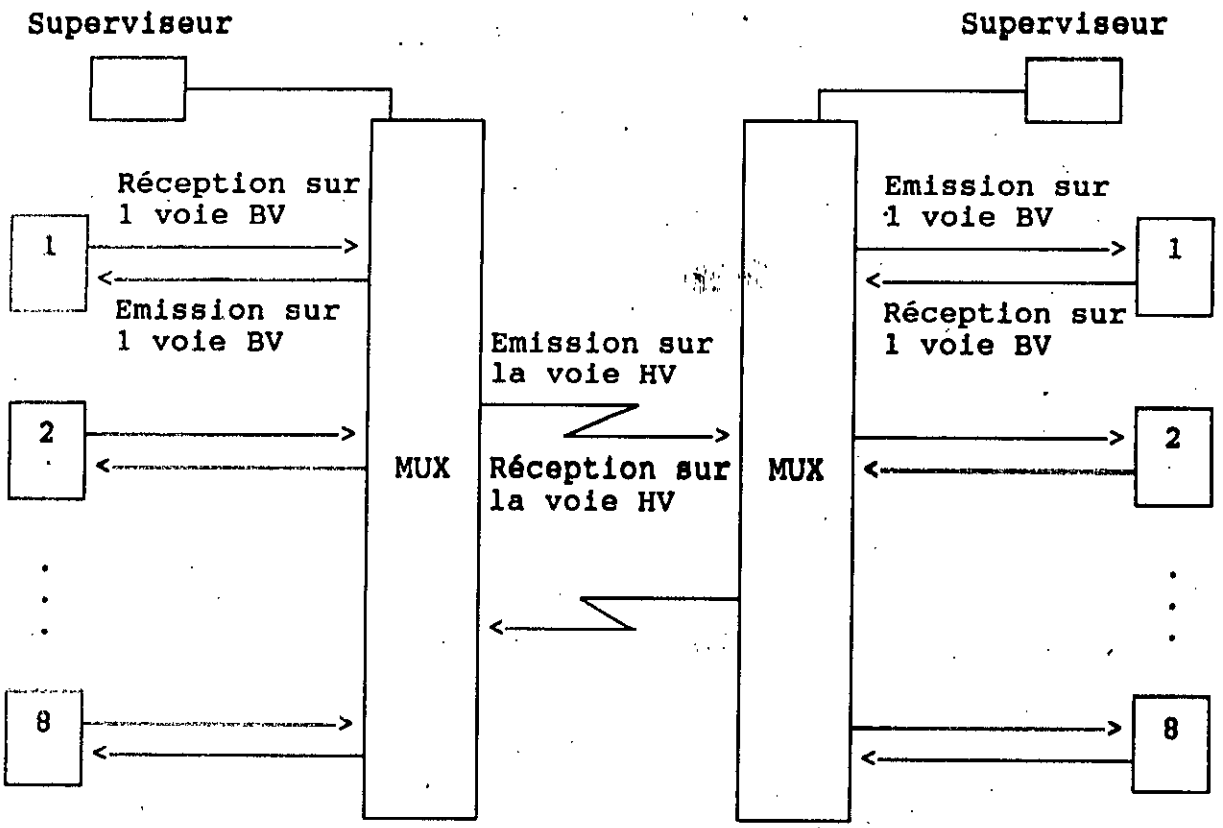


Figure 4.4. Circuits de parcours d'un multiplexage

— un processus d'émission sur la voie haute vitesse qui récupère l'information de la mémoire tampon de réception pour la véhiculer sur la voie haute vitesse;
— un processus de réception sur la voie haute vitesse qui récupère l'information émise sur la voie haute vitesse pour la diriger vers la mémoire tampon d'émission du périphérique destinataire;
— huit processus d'émission sur une voie basse vitesse qui récupèrent l'information de la mémoire tampon d'émission pour l'envoyer au périphérique destinataire.

Ces processus traduisent le service des périphériques connectés aux circuits précités.

Quant au terminal superviseur local, qui entre autres fonctions, établit les liaisons de communication, son service sera assuré par le processus gestion du terminal superviseur.

Une autre activité importante, que nous avons omis de citer jusqu'à présent, vient s'ajouter aux précédentes. Elle consiste

en la gestion de la file d'attente des messages, (appelés trames), prêts à véhiculer sur la voie haute vitesse, selon le principe «premier arrivé, premier servi». Cette activité sera divisée en un ensemble de huit (8) processus permettant chacun la constitution d'une trame relative à une voie basse vitesse et son enfilement dans la file d'attente. Ce sont les processus: constitution de la trame d'une voie basse vitesse.

2.1.2.- Organisation des processus

Les processus de notre système, notamment ceux liés au service des périphériques, se déroulent de manière simultanée; ce sont des processus parallèles. Cependant, l'unicité du processeur nous contraint à les traiter séquentiellement. On parvient toutefois, au moyen des interruptions matérielles (voir le paragraphe 3 du chapitre 5) à les traiter de manière pseudo-parallèle. Mais tous les processus ne peuvent pas être traités par interruption matérielle. Des dispositifs matériels doivent être disponibles pour permettre ce traitement. Cette condition est remplie pour les processus liés au service des périphériques. Ce n'est pourtant pas le cas des processus: constitution de la trame d'une voie basse vitesse qui devront être traités de manière séquentielle. La question qui se pose alors est la suivante: faut-il utiliser ces interruptions, et si oui, pour quels types de processus?

2.1.2.1.- Contrainte de temps imposée par le transfert d'un caractère

Le cycle de traitement des processus du système doit se faire à une fréquence suffisante pour permettre au processeur de traiter à temps les différents périphériques, et de déclencher, le cas échéant, les routines de service adéquates.

Les débits entrants et sortants des voies haute et basse vitesse du multiplexeur, constituent les contraintes qui définissent le temps maximum qu'on peut accorder au service de chaque périphérique. Cet intervalle de temps sera associé au chemin le plus long dans l'exécution d'un programme de service. C'est le chemin critique.

Lorsqu'un buffer de ligne demande un transfert de caractère

(chargement ou déchargement), l'intervalle de temps maximum pendant lequel la requête pourra rester en attente ne doit pas excéder le débit de l'information de ce périphérique:

$$Tw \leq tr \qquad (4.1)$$

où:

Tw = temps d'attente global (temps de service compris)
tr = temps de transfert d'un caractère sur la voie

Dans le cas contraire, un débordement si l'on est en réception ou une sous charge si l'on est en émission risquent de se produire.

2.1.2.2.- Interruptions sur toutes les voies

Grâce aux interfaces de communication matériels utilisés (OCTAL-UART et DUSART), tous les processus liés au service des périphériques, à savoir:

— la réception sur une voie basse vitesse (8 processus),
— l'émission sur la voie haute vitesse (1 processus),
— la réception sur la voie haute vitesse (1 processus),
— l'émission sur une voie basse vitesse (8 processus).

peuvent être traités par interruption.

L'inconvénient de ce mode de traitement revient au nombre de tests élevé à effectuer pour retrouver la source (ou la cause) de l'interruption, quand l'entrée des interruptions est commune.

En effet, si nous procédons par interruption, le problème se posera notamment pour les seize (16) processus des voies basse vitesse. Seize processus, donc seize causes différentes pour une même entrée d'interruption (située sur le microprocesseur). La distinction entre ces causes se fera selon une boucle de test, comme suit:

— 1 test pour retrouver le premier processus,
— 2 tests pour retrouver le deuxième processus,

.
.
.

— 16 tests pour retrouver le seizième processus.

Ainsi, chaque fois qu'on servira les seize processus cent trente six tests (somme des tests précédents) seront effectués.

Par contre, comme les deux processus de la voie haute vitesse disposent d'une entrée d'interruption différente de celle des processus des voies basse vitesse, on pourra les traiter par interruption sans que cela alourdisse la gestion du système; trois tests seulement seront effectués pour sonder les deux processus.

D'autre part, si on pose:

Tshe = temps de service maximum pour le processus **émission sur la voie haute vitesse,**
Tshr = temps de service maximum pour le processus **réception sur la voie haute vitesse,**
Trh = temps de transfert d'un caractère sur la ligne haute vitesse,

la relation (4.1) S'écrit alors pour ces deux processus:

$$Tshe \leq Trh \qquad (4.2)$$

et

$$Tshr \leq Trh \qquad (4.3)$$

or:

Trh = 8/9600, le chiffre 8 étant le nombre de bits du caractère.
= 833.33 μs

d'où: $Tshe \leq 833.33\ \mu s$

et $Tshr \leq 833.33\ \mu s$

2.1.2.3.- Boucle de test sur toutes les voies

Si l'on considérait la contrainte (de nombreux tests pour retrouver la cause de l'interruption) posée par le traitement par interruption d'un grand nombre de processus, qui à la limite alourdirait la gestion du système, nous conviendrions qu'il serait indiqué de traiter les processus **qui s y prêtent**, de manière bouclée (séquentielle) quand bien même leur déroulement physique serait parallèle.

Le système sert plusieurs processus du système, dont certains réclament des services simultanés. Dans ce cas, entre deux requêtes successives d'un même périphérique, le laps de temps accordé au service de ce dernier doit comprendre en plus des services des processus séquentiels (processus de constitution de la trame d'une voie basse vitesse), le service des requêtes des autres périphériques.

Ce délai de service sera déterminé à partir de la relation (4.1) qui s'écrit alors:

$$N\ Tsbr + N\ Tsf + M\ Tshe + M\ Tshr + N\ Tsbe \leq Tr \qquad (4.4)$$

où:

N = nombre de voies basse vitesse fonctionnelles,
Tsbr = temps de service maximum du processus réception sur une voie basse vitesse,
Tsf = temps de service maximum du processus constitution d'une trame de voie basse vitesse,
M = un diviseur du débit de la voie haute vitesse pour obtenir le débit de la voie basse vitesse le plus élevé,
Tshe = temps de service maximum du processus émission sur la voie haute vitesse,
Tshr = temps de service maximum du processus réception sur la voie haute vitesse,
Tsbe = temps de service maximum du processus émission sur une voie basse vitesse,
Tr = temps de transfert d'un caractère sur la voie la plus rapide.

Dans notre cas, N est égal à 8 (N = 8), tr est le temps de transfert de caractère sur la voie haute vitesse (Tr = 8/9600), et par conséquent, M sera égal à 1 (M = 1). Si par ailleurs, nous supposons que les temps de service des différents processus sont à peu près égaux, nous obtenons de la relation (4.4):

$$26\ Tsi \leq 8/9600$$

d'où:

$$Tsi \leq 32\ \mu s$$

La durée Tsi de chacun des 26 processus doit donc être inférieure ou égale à 32 μs pour que la branche de programmation

la plus longue dans le chemin critique s'exécute sans entraves.

On conçoit facilement que cet intervalle de temps ne constitue pas une marge de manoeuvre confortable pour exécuter toutes les routines d'initialisation, de contrôle, de transfert et de mise à jour des pointeurs, auxquelles une demande de service peut donner lieu.

Aussi, pour s'affranchir de la contrainte de temps due au débit de la ligne haute vitesse, a-t-on séparé le traitement des données sur la ligne haute vitesse de celui des données sur les lignes basse vitesse.

2.1.2.4.- Boucle de test sur les voies basse vitesse

La ligne haute vitesse bénéficie d'un régime particulier de par l'importance du débit des informations qui y sont convoyées: la durée de transfert d'un caractère y est trop courte pour que le temps d'attente octroyé à ce transfert puisse inclure les services des autres périphériques. Pour lever cet obstacle on attribue une priorité plus grande à la ligne haute vitesse par rapport à l'ensemble des lignes basse vitesse en la traitant par interruption. Le balayage des voies se limitera donc aux lignes basse vitesse en émission puis en réception.

Ainsi, on se libère immédiatement de la contrainte de brièveté du temps de service imposée par la brièveté des temps de transit des informations sur la ligne haute vitesse. Désormais, les temps de service octroyés à la voie haute vitesse seront limités par la contrainte des relations (4.2) et (4.3), soit:

$$Tshe \leq 833.33\ \mu s$$

et

$$Tshr \leq 833.33\ \mu s$$

Quant au délai imposé par la contrainte des débits des voies basse vitesse, pour servir l'ensemble des processus, il sera tiré de la relation (4.4) où Tr sera non plus le temps de transfert d'un caractère sur la voie haute vitesse, mais celui de la voie basse vitesse la plus rapide (Tr = 10/vitesse de la ligne, le chiffre 10 étant égal au 8 bits du caractère plus 1 bit START et 1 bit STOP).

Par ailleurs, l'apparition d'interfaces de communication asynchrones avec buffer de réception quadruplé (FIFO de réception de trois caractères), vers les années 1985/86, rallonge de manière considérable ces délais de service: la contrainte de temps n'est plus traduite par l'intervalle de temps qui sépare l'arrivée de deux caractères, mais celui qui s'écoule entre l'arrivée du premier et du quatrième caractère. Le temps de service gagné sera exploité pour étendre le système à de nouvelles applications, telles que:

— somme des vitesses nominales des voies basse vitesse supérieure ou égale à la vitesse de la voie composite,
— supervision étendue (choix du destinataire, sortie de résultats statistiques,...),
— retransmission automatique en cas d'erreur,
— possibilité de liaisons locales,
— cascade de multiplexeurs,
— et autres.

Quant à La relation (4.4), elle s'écrit alors:

$$N\ Tsbr + N\ Tsf + M\ Tshe + M\ Tshr + N\ tsbe \leq 3\ Tr \qquad (4.5)$$

Pour illustrer ceci, nous considérerons les deux cas limites suivants:

— premier cas: Tr = 10/9600

Dans ce cas le débit de la voie basse vitesse la plus rapide est égal à celui de la voie composite. Une seule des huit voies basse vitesse est donc fonctionnelle, et la relation (4.5) s'écrit alors:

$$Tsbr + Tsf + Tshe + Tshr + tsbe \leq 3\ Tr$$

Si nous supposons que la durée de service est plus ou moins équivalente pour l'ensemble de ces services, nous obtenons:

$$5\ Tsi \leq 3 \times 10/9600$$

d'où:

$$Tsi \leq 625\ \mu s$$

ce qui représente une très confortable marge de temps pour l'exécution d'un processus.

— **deuxième cas:** Tr = 10/4800

Dans ce cas les huit voies basse vitesse peuvent être fonctionnelles. La relation (4.5) donne alors:

$$8\ Tsbr + 8\ Tsf + 2\ Tshe + 2\ Tshr + 8\ tsbe \leq 3\ Tr$$

Si l'on supposait, comme dans le cas précédent que les durées de service sont du même ordre, on obtiendrait:

$$28\ Tsi \leq 3 \times 10/4800$$

d'où:

$$Tsi \leq 223.21\ \mu s$$

ce qui constitue également, une marge de temps confortable pour le service d'un processus.

Ces chiffres ne concernent, bien sûr, que les chemins critiques. En imposant aux processus des temps de service inférieurs à ces valeurs, nous garantissons l'exécution de la branche la plus longue du chemin critique sans entrave aucune.

Enfin, il arrive que l'application soit tellement sophistiquée, qu'elle puise beaucoup de temps dans les Tsb. Dans ce cas, si le Tsb ne suffit plus à servir à temps les périphériques, un contrôle automatique des débits des voies basse vitesse est effectué: dès que la FIFO est pleine, un signal est généré (CTS haut), stoppant momentanément le débit de la voie. Dès qu'une position est libre dans la FIFO, le débit reprend (CTS bas). Cette caractéristique est une nouveauté pour les interfaces de communication.

Quant au terminal superviseur, sa fréquence d'intervention est négligeable devant celle des périphériques connectés aux voies basse vitesse. De ce fait nous pouvons, indépendamment, le traiter par interruption ou par boucle de test. Comme son intervention n'est pas prioritaire, nous avons opté pour l'alternative de le traiter par boucle de test.

_ **RECAPITULATIF**_

De cette organisation découle un parallélisme entre les processus:

— Emission sur la voie haute vitesse (1 processus),
et — Réception sur la voie haute vitesse (1 processus),
et — Gestion du terminal superviseur (1 processus),
ou — Réception sur une voie basse vitesse (1 processus),
ou — Constitution de la trame d'une voie basse vitesse, (1 processus)
ou — Emission sur une voie basse vitesse (1 processus).

Quant aux processus suivants, ils seront traités de manière séquentielle:

— Gestion du terminal superviseur (1 processus),
— Emission sur une voie basse vitesse (8 processus)
— Constitution de la trame d'une voie basse vitesse (8 processus)
— Réception sur une voie basse vitesse (8 processus).

Le décompte de tous ces processus donne trois (3) processus parallèles et vingt-cinq (25) processus séquentiels.

2.2.- MODELISATION DU SYSTEME

Le processus de modélisation est très complexe et le degré de sa difficulté croît avec la complexité du système à étudier. Les principaux facteurs qui rendent cette tâche ardue concernent:

— la difficulté de représentation de certaines composantes du système,
— une grande variabilité dans le comportement du système,
— l'interprétation de certains facteurs régis par l'élément humain où le subjectif prime sur le rationnel,
— la détermination des facteurs extérieurs qui affectent, par leur variabilité, le comportement du système.

La démarche suivie lors de cette modélisation présente l'alternative suivante:

— démarrer d'un modèle simplifié ne représentant que grossièrement le système, puis le développer et l'améliorer jusqu'à obtention d'une représentation fidèle du système réel.
— diviser le système en sous-systèmes, modéliser ces derniers un à un et séparément pour les combiner et aboutir au modèle final.

Il n'existe pas de règle préétablie pour décider de la meilleure alternative. Le choix est laissé à la perspicacité du concepteur.

Pour gérer de manière efficace notre système, nous avons pensé le décomposer en processus. De plus, nous avons organisé ses processus de la façon qui répond le mieux aux caractéristiques du matériel utilisé, tenant compte des nouveautés introduites. Ces décompositions et organisations du système facilitent de manière considérable sa modélisation; il suffira d'élaborer le modèle de chaque processus, puis établir les liens qui réunissent l'ensemble des processus modélisés selon l'organisation à laquelle ils ont été soumis dans le paragraphe 2.1.2.

Nous noterons cependant, que ces processus traduisent des activités plutôt logicielles. Or, la modélisation du système nécessitera quelques fonctions matérielles, notamment l'arrivée des données dans les FIFO de réception.

La modélisation, que nous allons effectuer sera discrète (système discret à entités discrètes) et dans un souci de flexibilité, elle se fera par processus.

2.2.1. Modélisation des arrivées

L'information débitée par un périphérique connecté à une voie basse vitesse, arrive dans une FIFO de réception de trois positions mémoires (3 caractères) qui la stocke jusqu'à ce que le processus émission sur une voie basse vitesse se charge de son acquisition.

Pour modéliser ceci, SLAM II nous offre les moyens suivants:
— l'arrivée d'une entité «caractère» sera spécifiée par le symbole CREATE,
— son stockage dans la FIFO de réception sera traduit par le symbole QUEUE.

Les arrivées sur le terminal superviseur sont à quelques détails près similaires aux arrivées sur les périphériques connectés aux voies basse vitesse. Lorsqu'une entité «caractère» est créée, elle est stockée dans une file de trois positions mémoires (c'est la FIFO de réception du superviseur) jusqu'à ce que le processus gestion du superviseur, se charge de son acquisition.

2.2.2.- Modélisation du processus gestion du superviseur

Le processus gestion du superviseur est le premier processus d'une suite de processus traités par boucle de test (séquentiellement). Son traitement correspondra donc à la création d'une nouvelle entité qu'on appellera «boucle de test».

Une fois l'entité créée, le processus est déclenché. Le contenu de la file 9 est testé pour savoir si une requête de connexion est émise. Dans le cas affirmatif, la liaison est établie et une éventuelle reconfiguration et réalisée. Le temps écoulé dans ces opérations est représenté par l'activité de service 11. Dans le cas où la file 9 est vide le temps écoulé dans le test sera représenté par l'activité de service 12.

Cependant, sachant qu'une activité de service requiert un serveur, qui dans notre cas est unique (le processeur), ces activités ne peuvent avoir lieu que si le processeur est libre. L'entité créée doit donc attendre la libération de la ressource «processeur», dans la file représentée par le symbole (ou node) AWAIT. Une fois l'activité achevée, la ressource est libérée à l'aide du node FREE.

2.2.3.- Modélisation des processus réception sur une voie basse vitesse

Ces huit (8) processus sont inclus dans la boucle de test. Ils s'enchaînent au processus gestion du superviseur.

Rappelons qu'ils consistent à décharger la FIFO de réception dans la mémoire tampon de réception correspondante. Pour cela, la queue i, i = 1 à 8, dont la capacité maximum est de trois caractères est testée. Si elle se trouve pleine, son contenu (3 caractères) est déchargé dans la file 1i à l'aide de l'événement discret EVENT 2; sinon, si elle n'est pas vide (1 ou 2 caractères) une entité y est prise puis stockée dans la file 1i, à l'aide de l'événement discret EVENT 1. Les temps écoulés dans ces deux manoeuvres seront respectivement représentés par les activités de service 11 et 1. Dans le cas où la queue est vide, le temps écoulé à effectuer le test sera représenté par l'activité de service 2.

Pour exécuter ces activités, le serveur doit être au préalable

libre. En attendant sa libération, l'entité «boucle de test» doit patienter dans une file du type AWAIT. Lorsque l'activité est achevée, le serveur est libéré à l'aide du node FREE.

Ces huit (8) processus dont le traitement est identique, sont traités de manière bouclée.

L'événement EVENT 1, consiste à:
— puiser une entité de la file i, à l'aide du sous-programme Fortran, RMOVE (NRANK, IFILE, A) résidant dans la librairie SLAM. NRANK correspond au rang de l'entité dans la file. IFILE désigne le numéro de la file et A est un vecteur destiné à contenir les attributs de l'entité;
— la stocker dans la file 1i, à l'aide du sous-programme Fortran, FILEM (IFILE, A), résidant dans la librairie SLAM. IFILE correspond au numéro de la file et A désigne le vecteur destiné à contenir les attributs de l'entité.

Cet événement est donc une procédure Fortran, appelant les deux sous-programmes ci-dessus.

Quant à l'événement EVENT 2, ce n'est rien d'autre qu'une triple itération sur l'événement EVENT 1.

2.2.4.- Modélisation des processus constitution de la trame d'une voie basse vitesse

Ces huit (8) processus font partie de la boucle de test. Ils s'enchaînent aux processus émission sur une voie basse vitesse.

Rappelons qu'ils consistent à enfiler une trame d'une voie i dans une file d'attente des trames prêtes à être envoyées sur la voie haute vitesse.

Pour effectuer ceci, on teste le contenu de la file 1i (mémoire tampon de réception). S'il a atteint la longueur d'une trame (une longueur fixe), une nouvelle entrée dans la file d'attente des trames (file 19) est effectuée, à l'aide de l'événement discret EVENT 3. Le temps écoulé dans cette manoeuvre est représenté par l'activité de service 3. Dans le cas où le contenu de la file 1i est inférieur à la longueur d'une trame, le temps écoulé à faire le test est représenté par l'activité de service 4.

Bien sur, avant d'entamer toute activité, l'entité «boucle de test» doit attendre la libération de la ressource «processeur». Et, à la fin des activités, la ressource est libérée.

2.2.5.- Modélisation du processus émission sur la voie haute vitesse

Ce processus est traité par interruption. La fréquence de son exécution et directement liée au débit de la voie haute vitesse en émission: à chaque fois que l'interface de communication (le DUSART) est prêt à émettre un nouveau caractère, une nouvelle entité est créée; nous appellerons cette entité «Interruption d'émission». Elle sera introduite à l'aide du noeud CREATE, XX(19),,,,1; où XX(19) correspond au débit de la voie haute vitesse en émission, soit 9600 bit/s ou un caractère toutes les 833.33 μs d'où XX(19) = 833.33 μs.

Rappelons que la fonction de ce processus est de prendre une entité de la trame de tête de file pour l'émettre sur la voie haute vitesse.

Donc, lorsqu'une entité «interruption d'émission» est créée, le contenu de la file d'attente 19 est testé. S'il se trouve non nul, une entité «caractère» est prise de la file 1i correspondant à la source de la trame, puis stockée dans la file 20, et un attribut relatif à la destination de la trame est associé à ce caractère. Ceci est réalisé à l'aide de l'événement EVENT 4. Le temps écoulé dans cette tâche est représenté par l'activité de service 5. Dans le cas où la file 19 est vide, le temps écoulé dans le test est représenté par l'activité de service 6.

Hormis les caractères du message de la trame, chaque nouvel envoi de trame débutera par l'émission de deux caractères de contrôle. Lorsqu'une trame est en cours d'émission, elle est défilée de la file 19 à l'aide de l'événement discret EVENT 5.

La priorité accordée à ce processus de par son déclenchement par interruption, lui octroie le droit d'enlever la ressource «processeur» qui est le serveur, à toute entité qui la détenait, pour la mettre au service des activités 5 ou 6. Ceci est réalisé à l'aide du node PREEMPT.

Une fois l'activité de service achevée, le serveur est libéré avec le node FREE. Il retourne alors à l'entité à laquelle il a été enlevé.

2.2.6.- Modélisation du processus réception sur la voie haute vitesse

Ce processus est traité par interruption. La fréquence de son exécution est directement liée au débit de la voie haute vitesse en réception: à chaque fois que l'interface de communication (le DUSART) est prêt à recevoir un caractère, une nouvelle entité est créée. Nous appellerons cette entité «interruption de réception». Elle sera introduite à l'aide du noeud CREATE,XX(20),,,,1; où XX(20) correspond au débit de la voie haute vitesse en réception soit 9600 bits/s, d'où XX(20) = 833.33 μs.

Rappelons que la fonction de ce processus est de récupérer une entité «caractère» de la ligne haute vitesse pour la stocker dans la mémoire tampon de réception du destinataire.

De ce fait, la file 20 est testée. Si son contenu est non nul, une entité «caractère» en est défilée et ses attributs sont placés dans le vecteur ATRIB, à l'aide de l'événement EVENT 6. A la sortie de ce node, l'entité courante n'est donc plus l'entité «interruption de réception» mais plutôt l'entité «caractère». Après test sur sa provenance, contenue dans l'attribut ATRIB(1), cette entité est dirigée vers la file 2i, i= 1 à 8, destinataire, symbolisée par un node AWAIT avec le numéro de la file comme attribut. Les deux caractères de contrôle de chaque trame sont bien entendu, écartés. Le temps écoulé dans toutes ces opérations est représenté par l'activité de service 7. Dans le cas où la file 40 est vide, le temps de service est désigné par l'activité 8.

Ce processus est prioritaire à tous les autres processus. Il peut donc s'accaparer le serveur pour ses propres activités. Ceci, est réalisé à l'aide du node PREEMPT. Sa priorité par rapport au processus émission sur la voie haute vitesse est établie par l'ordre des files ayant acquisition à la ressource «processeur», lors de la déclaration de cette ressource: la première file mentionnée est la plus prioritaire.

Une fois l'activité achevée, le serveur est libéré.

2.2.7.- Modélisation des processus émission sur une voie basse vitesse

Ces huit (8) processus appartiennent à la boucle de test. Ils s'enchaînent aux processus constitution de la trame d'une voie basse vitesse.

Rappelons que leur traitement consiste à émettre un caractère de la mémoire tampon d'émission sur la voie basse vitesse destinataire.

Pour réaliser ceci, le contenu de la file 2i (mémoire tampon d'émission) est testé. S'il est non nul, une entité «caractère» y est puisée, à l'aide de l'événement EVENT 7. Le temps mis dans cette opération est représenté par l'activité de service 9. Dans le cas où la file 2i est vide, le temps écoulé dans le test est désigné par l'activité de service 10.

Il va de soi, qu'avant d'entamer toute activité, l'entité «boucle de test» doit attendre la libération de la ressource «processeur». A la fin de l'activité, cette ressource est libérée.

2.2.8.- Caractéristiques du modèle final

La combinaison de l'ensemble des sous-modèles que nous avons élaboré, constitue le modèle final. Les caractéristiques de ce modèle sont les suivantes:

- modélisation discrète,
- modélisation par processus,
- un serveur unique: le processeur,
- quatre types d'entités:
  - l'entité caractère,
  - l'entité boucle de test,
  - l'entité interruption d'émission,
  - l'entité interruption de réception,
- un attribut pour l'entité caractère,
- douze activités de service,
- dix-neuf files du type QUEUE,
- neuf files du type AWAIT,
- deux files du type PREEMPT,
- sept événements discrets EVENT.

2.3.- Collecte et analyse des données [20]

Une opération importante dans le processus de toute simulation consiste en la collecte et l'analyse des données. Cette fonction est requise aussi bien pour définir les entrées du modèle que pour mesurer les performances de l'expérimentation sur ce modèle. En effet, sans données fiables, les résultats de la simulation se trouvent inévitablement erronés et une analyse de ces derniers devient hors de propos.

La première tâche importante à effectuer lors d'une collecte des données, consiste à recenser les paramètres d'entrée à évaluer. La liste exhaustive de ces paramètres pour notre modèle est la suivante:
— les inter-arrivées pour les 8 voies basse vitesse à l'émission,
— les inter-arrivées pour la voie haute vitesse à l'émission et à la réception,
— les inter-arrivées pour le superviseur,
— les temps de service de tous les processus que nous avons déjà déterminés.

Après identification des paramètres à collecter, il ne restera plus qu'à les acquérir, puis si besoin est, ajuster leur évolution à des lois probabilistes.

Différentes méthodes permettent l'acquisition des données. Dans certains cas, l'information requise est disponible dans des documents qu'il reste à localiser, puis y accéder. Dans d'autres cas, elle nécessite l'utilisation de questionnaires et d'expérimentation physique.

Dans notre présente étude, nous procéderons à deux types d'acquisition des données relatives aux temps de service des processus:
— la première se fera sur un avis d'expert. Elle sera destinée à tester le modèle,
— la seconde sera relevée sur le système réel. Elle servira à la validation et à l'analyse du modèle.

Les inter-arrivées sur les voies basse et haute vitesse seront directement déterminées à partir des débits de ces voies, soit:
— 10 bits/(vitesse de la ligne) bit par seconde, pour les 8 voies basse vitesse, en émission;

— 8 bits/9600 bits par seconde, pour la voie haute vitesse en émission et en réception.

Quant aux inter-arrivées sur le superviseur, elles seront ajustées à une loi exponentielle.

2.3.1. Acquisition des services sur avis d'expert

Cette acquisition de données, aura pour but d'exécuter dans un premier temps notre simulation, tester et vérifier notre modèle puis tirer les premiers résultats qui nous intéressent.

Nous avons pu déterminer, dans le paragraphe 2.1. la durée de service maximum d'un processus, imposée par le débit de la voie basse vitesse la plus rapide. Cette durée a été évaluée à 223 μs.

D'autre part, nous avions bien vu, lors de la modélisation du système, que ces services comprenaient deux tâches distinctes:
— la première déterminait si un lancement du service proprement dit, devait avoir lieu ou non; nous l'appellerons «service préliminaire»;
— la seconde comprenait le service proprement dit ou «service» tout simplement.

Dans notre modèle, les services 2, 4, 6, 8, et 10 sont des services préliminaires. Quant aux services 1, 3, 5, 7, et 9 ils regroupent chacun, un service préliminaire suivi d'un service.

Nous retiendrons donc, pour les services 1, 3, 5, 7, et 9 la durée de 200 μs, sachant qu'une moyenne de 4 μs par instruction donne 50 instructions pour un processus. Quant aux services préliminaires, ils seront évalués à une vingtaine d'instructions, ce qui devrait constituer une marge de manoeuvre confortable pour exécuter tous les tests qui précèdent une éventuelle mise en service. A raison d'une moyenne de 4 μs par instruction, nous obtenons une durée de 80 μs pour un service préliminaire.

2.3.2. Acquisition des services sur système réel

Cette acquisition des données, servira tout d'abord, à la validation du modèle, puis sera exploitée pour l'analyse des

sorties de ce modèle.

Dans cette méthode, les temps de service des processus, qui réellement, correspondent aux durées d'exécution des procédures traduisant ces processus, sont mesurés sur la machine de développement du système réel, ce dernier étant réalisé dans un premier temps par émulation. Ces durées de services sont présentées dans le tableau ci-dessous:

Tableau 4.1.:

service	Désignation	Temps de Service
1	Tsbr	279 µs
2	Tsbr'	120 µs
3	Tsf	239 µs
4	Tsf'	60 µs
5	Tshe	230 µs
6	Tshe'	108 µs
7	Tshr	141 µs
9	Tsbe	179 µs
10	Tsbe'	48 µs
11	Tsbr3	354 µs

2.3.3.- Ajustement d'une loi exponentielle aux inter-arrivées du superviseur

La fréquence des arrivées sur le superviseur présente une allure poissonienne. Ceci nous a amené à penser ajuster une loi exponentielle aux durées des inter-arrivées observées sur le superviseur. Cet ajustement s'est effectué comme suit:

1- Estimation du temps moyen entre deux arrivées, $1/\lambda$

$$1/\lambda = \frac{\sum_{i=1}^{K} n_i x_i}{N} \qquad (4.6)$$

Dans notre cas, k=5 et N=8, voir tableau 4.2, d'où:

$$1/\lambda = \frac{\sum_{i=1}^{5} n_i x_i}{8} \simeq 41.25 \text{ min}$$

d'où $\lambda = 0.024$

2- Test du X^2

2.1- Construire l'hypothèse nulle H_0

H_0 : La loi de distribution est exponentielle contre l'hypothèse alternative H_1

H_1 : non H_0

2.2- Etablir le nombre d'observations N1 et le seuil de l'erreur α

— $N1 = k - 1 = 4$

— $0.01 \leq \alpha \leq 0.05$

2.3- Déterminer les fréquences théoriques $N\ p_i$

$$P_i = \int_{x1}^{x2} \lambda e^{-\lambda x}\, dx = e^{-\lambda x1} - e^{-\lambda x2} \qquad (4.7)$$

Tableau 4.2.

x_i (min)	n_i	$N\ P_i$
$0 \leq t \leq 30$	4	4.08
$30 \leq t \leq 60$	2	2
$60 \leq t \leq 90$	1	1.04
$90 \leq t \leq 120$	1	0.48
$t > 120$	0	0.45
	$N = 8$	

2.4- Calculer la statistique:

$$Z = \sum_{i=1}^{k} \frac{(n_i - N\ P_i)^2}{N\ P_i} = 1.013 \qquad (4.8)$$

2.5- Décider en fonction de:

$Z \leq X^2_{k-1}(\alpha)$ ⟹ Accepter H_0 avec un risque d'erreur de α

$Z > X^2_{k-1}(\alpha)$ ⟹ Rejeter H_0 avec un risque d'erreur de α

Dans notre cas:

$$X^2_4(0.10) = 1.06$$

donc:

$$Z < X^2_4(0.10) \Longrightarrow H_0 \text{ acceptée}$$

Ainsi, nous pouvons ajuster une loi exponentielle de paramètre $\lambda = 0.024$ aux inter-arrivées sur le superviseur avec un risque d'erreur de premier type de 1 %.

2.4.- VERIFICATION ET VALIDATION DU MODELE [18]

L'un des problèmes cruciaux auxquels fait face un simulateur du «monde réel» est celui de déterminer si le modèle de simulation est une représentation fidèle du système étudié. Cependant l'éventail littéraire dans le domaine de la validation ([25], [26], [27], [28] et [29]) reste très peu étendu, et ce qui fut développé jusqu'à présent est plutôt de nature philosophique que sous la forme de recommandations pratiques.

Pour dissiper toute confusion entre les sens de vérification et de validation, il conviendrait de commencer par une définition simple de ces termes. La vérification détermine si le modèle fonctionne comme prévu, c'est-à-dire, si le programme de simulation traduit correctement le modèle élaboré. Quant à la validation, elle détermine si le modèle (par opposition au programme de simulation) est une représentation précise du système étudié (par opposition au modèle élaboré).

2.4.1.- Vérification du modèle

Plusieurs techniques sont utilisées pour la mise au point (ou debugging) d'un programme de simulation. Nous présenterons dans ce qui suit celles que nous avons utilisées pour la mise au point de notre programme de simulation.

Technique 1

Pour procéder à une mise au point rapide et aisée du programme de simulation, il est vivement conseillé de partir d'une représentation très simplifiée du système réel, la faire fonctionner correctement puis, au fur et à mesure, la perfectionner à l'aide de nouveaux tests et de nouvelles fonctions sous forme de procédures, jusqu'à obtention du modèle représentatif final; en effet, exécuter un programme de simulation entier, non testé, se fera très probablement non sans erreurs, et la localisation de ces erreurs devient tâche extrêmement difficile dans un programme de si grande taille.

La décomposition de notre système en processus facilita de manière considérable l'application de cette technique. Nous avons tout d'abord procédé à la mise au point des processus qui pouvaient être traités indépendamment des autres, puis au fur et à mesure, nous avons relié tous les processus du système. Une fois que le modèle simplifié mais entier fonctionnait correctement, nous nous sommes occupés de son ajustement au système réel en y introduisant les niveaux de détail et les modifications qui s'imposaient.

Technique 2

L'une des techniques les plus performantes dans la mise au point d'un programme de simulation consiste à exécuter pas à pas (Trace) ce dernier, et à consulter à chaque pas le contenu des registres d'état et des paramètres statistiques du modèle pour vérifier si l'évolution du programme se déroule comme.prévu.

Souvent, pour pouvoir suivre l'évolution du programme aussi loin que possible, nous nous trouvons dans l'obligation de réduire au maximum la longueur des boucles de traitement du modèle, et par conséquent, contraints de tester des modèles réduits. Dans notre cas, nous avons effectué le «trace» sur une longueur de la trame égale à trois.

Technique 3

Cette technique nous a permis de vérifier la cohérence des résultats de la simulation. Cette vérification a porté sur deux éléments:

1- la cohérence dans l'évolution des entités dans le système. En effet, on constate que le nombre d'arrivées, le nombre d'entités ayant subi une activité i et le contenu courant des files sont cohérents, si l'exécution s'est faite sans erreurs;

2- la cohérence dans les temps d'attente des FIFO de réception. Nous vérifions bien que ces temps d'attente sont nettement inférieurs, en dehors des cas limites, à la fréquence des arrivées sur les voies correspondantes.

2.4.2.- Validation du modèle

Nous nous sommes conformés, pour la validation de notre modèle à l'approche en trois phases, développée par Naylor et Finger [26].

Phase 1

L'objectif principal de la première phase de la validation est l'élaboration d'un modèle qui, à première vue, paraît valable aux experts du système. Pour ce faire, le modéliste doit se référer:
— aux observations et recommandations des experts du système,
— à la théorie et aux connaissances déjà développées et accumulées dans le domaine,
— à sa propre intuition.

Comme dans notre cas, la construction du système est une tâche qui nous incombe, nous constituons, de ce fait, les experts de ce système.

Phase 2

L'objectif de la deuxième phase de la validation est de tester quantitativement les hypothèses construites lors de l'élaboration du modèle. L'un des outils les plus utilisés lors de cette étape de la validation est l'analyse sensitive. Cette analyse peut porter sur l'étude de la variation de la sortie du modèle en fonction d'un léger changement dans un paramètre d'entrée. Si la sortie est particulièrement sensible à un quelconque paramètre, une meilleure estimation de ce paramètre doit être effectuée. Un autre champ d'application de l'analyse sensitive apparaît dans la détermination du niveau de détail à introduire dans chaque bloc du système. Il ne s'agit ni de trop détailler l'application au point d'alourdir inutilement le modèle, ni d'omettre des détails importants au risque d'altérer la sortie de ce modèle. Le tout est de bien cerner le problème et ses objectifs, et d'octroyer à chaque composante du système, le niveau de détail qu'il faut.

Phase 3

Enfin, la phase la plus déterminante dans la validation d'un modèle, est celle qui établit que la sortie de celui-ci est très proche de celle du système réel, si celui-ci existe. Un nombre de tests statistiques a été suggéré dans la littérature de la validation, pour procéder à cette comparaison. Seulement, la non stationnarité et l'autocorrélation, aussi bien des sorties du système réel que de celles du modèle, font que les tests statistiques classiques, basés sur les observations indépendantes et identiquement distribuées (IID) ne sont plus directement applicables. Aussi, les trois approches statistiques appropriées, les plus souvent utilisées sont:

— **l'approche par inspection** qui consiste à calculer les statistiques du modèle, celles du système réel, puis les comparer sans recourir à une procédure statistique formelle. Cette approche est souvent utilisée pour des comparaisons où le nombre d'observations est réduit;

— **l'approche par intervalle de confiance** qui consiste à construire, un intervalle de confiance sur la différence des moyennes des variables aléatoires indépendantes relatives au système réel et au modèle;

— **l'approche par séries temporelles** qui identifie les observations à des séries chronologiques et les analyse selon des techniques économétriques. L'analyse spectrale procède par l'estimation de la transformée de Fourrier de la fonction d'autocovariance, puis la construction de l'intervalle de confiance de la différence des logarithmes du spectre. Voir pour un complément d'information les ouvrages [23], [24] et [25].

Nous avons appliqué pour la validation de notre modèle, l'approche par inspection. L'approche par intervalle de confiance ne peut lui être appliquée, du fait qu'il est exécuté sous sa forme déterministe. Cependant, envisageant son exécution sous sa forme stochastique, sur un système très rapide, il conviendrait d'exposer, succinctement, le principe de cette approche.

2.4.2.1.- Approche par inspection

Pour parvenir à une bonne comparaison entre le modèle élaboré et le système réel réalisé, nous nous sommes conformés à la technique qui préconise leur comparaison dans un environnement statistique similaire, afin de réduire la variance de la différence des moyennes des variables aléatoires des deux objets de la comparaison et aboutir à des résultats beaucoup plus fiables [26].

En effet, nous avons utilisé comme entrée du modèle, des données extraites du système réel. Nous avons exécuté notre simulation et notre réalisation sur des laps de temps équivalents et nous avons essayé autant que possible, de démarrer nos tests sous les mêmes conditions de départ. Seulement, le lancement du système réel, qui est fonction d'un lancement manuel des périphériques qui lui sont connectés, reste toujours sujet à une variabilité

— **l'approche par inspection** qui consiste à calculer les statistiques du modèle, celles du système réel, puis les comparer sans recourir à une procédure statistique formelle. Cette approche est souvent utilisée pour des comparaisons où le nombre d'observations est réduit;

— **l'approche par intervalle de confiance** qui consiste à construire, un intervalle de confiance sur la différence des moyennes des variables aléatoires indépendantes relatives au système réel et au modèle;

— **l'approche par séries temporelles** qui identifie les observations à des séries chronologiques et les analyse selon des techniques économétriques. L'analyse spectrale procède par l'estimation de la transformée de Fourrier de la fonction d'autocovariance, puis la construction de l'intervalle de confiance de la différence des logarithmes du spectre. Voir pour un complément d'information les ouvrages [23], [24] et [25].

Nous avons appliqué pour la validation de notre modèle, l'approche par inspection. L'approche par intervalle de confiance ne peut lui être appliquée, du fait qu'il est exécuté sous sa forme déterministe. Cependant, envisageant son exécution sous sa forme stochastique, sur un système très rapide, il conviendrait d'exposer, succinctement, le principe de cette approche.

2.4.2.1.- Approche par inspection

Pour parvenir à une bonne comparaison entre le modèle élaboré et le système réel réalisé, nous nous sommes conformés à la technique qui préconise leur comparaison dans un environnement statistique similaire, afin de réduire la variance de la différence des moyennes des variables aléatoires des deux objets de la comparaison et aboutir à des résultats beaucoup plus fiables [26].

En effet, nous avons utilisé comme entrée du modèle, des données extraites du système réel. Nous avons exécuté notre simulation et notre réalisation sur des laps de temps équivalents et nous avons essayé autant que possible, de démarrer nos tests sous les mêmes conditions de départ. Seulement, le lancement du système réel, qui est fonction d'un lancement manuel des périphériques qui lui sont connectés, reste toujours sujet à une variabilité

dans les conditions de départ, notamment, celles concernant le moment exact du lancement d'un périphérique.

Ainsi, nous avons exploité cette légère variabilité pour obtenir une série de résultats du système réel et une série de résultats du modèle. Une série de résultats se présente sous la forme d'une suite de moyennes, de la variable étudiée, obtenue à la suite d'exécutions indépendantes. Les moyennes des variables du modèle sont calculées par le logiciel de simulation. Celles du système réel, sont calculées à l'aide d'une procédure de ce dernier.

Nous pouvons ainsi, comparer toutes les variables du modèle à celles du système réel. Pour une contrainte de temps et de coût, nous nous sommes juste intéressés à la variable «taille mémoire tampon de réception», qui constitue notre préoccupation principale.

Les résultats obtenus à la suite de 10 exécutions du système et du modèle sur la variable «taille mémoire de réception de la voie 1», sont fournis par la tableau 4.3.

Tableau 4.3.

Test	Moyenne du système X_i	Moyenne du modèle Y_i	$\frac{\lvert X_i - Y_i \rvert}{X_i}$
1	60.28	59.72	0.9 %
2	56.55	59.17	4.6 %
3	63.69	60.38	5.2 %
4	62.99	60.50	3.9 %
5	60.59	60.10	0.8 %
6	58.22	59.85	2.8 %
7	59.98	60.02	0.1 %
8	61.62	60.12	2.4 %
9	60.04	59.52	0.8 %
10	59.10	60.02	1.5 %

Après comparaison de ces résultats, nous constatons que la différence entre le système et le modèle est très faible.

2.4.2.2. - Approche par intervalle de confiance

Pour comparer un modèle au système réel, nous convenons qu'il est plus judicieux de construire un intervalle de confiance plutôt

que tester une hypothèse. En effet, l'hypothèse nulle selon laquelle le système réel et le modèle sont identiques, est rejetée d'office, du moment où le modèle n'est qu'une approximation du système. Si l'on doit donc construire une hypothèse, ce sera plutôt pour tester si la différence entre le modèle et le système est suffisamment significative pour affecter les résultats du modèle. Or cette information augmentée du degré de différence entre les deux objets de la comparaison, sont systématiquement fournis par la construction d'un intervalle de confiance.

Soit X_i la moyenne des observations du système réel à la 1ère réplique. Les X_i sont des variables IID. Et soit Y_i la moyenne des observations du modèle à la ième exécution. Les Y_i sont aussi des variables IID. Selon certains critères, l'une des deux approches suivantes, de construction de l'intervalle de confiance est utilisée.

1- Intervalle de confiance «paired-t»

Cette approche est utilisée lorsque le nombre d'observations est égal chez les deux partis. Elle présente l'avantage de ne point exiger l'indépendance entre les X_i et les Y_i, ni la normalité de la distribution des X_i et des Y_i.

Si $Z_i = X_i - Y_i$ pour $i = 1$ à n

nous avons:

$$\bar{Z}(n) = \frac{\sum_{i=1}^{n} Z_i}{n} \quad \text{et} \quad \hat{\sigma}^2[\bar{Z}(n)] = \frac{\sum_{i=1}^{n} [Z_i - \bar{Z}(n)]}{n(n-1)}$$

et l'intervalle de confiance à 100 $(1-\alpha)$ pour cent s'exprime:

$$\bar{Z}(n) \pm t_{n-1,1-\alpha/2} \sqrt{\hat{\sigma}^2[\bar{Z}(n)]}$$

2- Intervalle de confiance de Welch

La contrainte d'égalité du nombre d'observations sur le modèle et le système n'est pas imposée par cette approche. Par contre, l'indépendance entre les variables X_i et Y_i et leur normalité de distribution doivent être respectées.

soit: $\bar{X}(n1) = \frac{\sum_{i=1}^{n1} Z_i}{n1}$ et $\bar{Y}(n2) = \frac{\sum_{i=1}^{n2} Y_i}{n2}$

et

$S_1^2(n1) = \frac{\sum_{i=1}^{n1} [X_i - \bar{X}(n1)]^2}{n1-1}$ et $S_2^2(n2) = \frac{\sum_{i=1}^{n2} [Y_i - \bar{Y}(n2)]^2}{n2-1}$

le degré de liberté estimé $\hat{f}$ est calculé comme suit:

$$\hat{f} = \frac{[S_1^2(n1)/n1 + S_2^2(n2)/n2]^2}{[S_1^2(n1)/n1]^2/(n1-1) + [S_2^2(n2)/n2]^2/(n2-1)} \qquad (4.9)$$

et l'intervalle de confiance est:

$$[\bar{X}(n1) - \bar{Y}(n2)] \pm t_{\hat{f},1-\alpha/2} \left[\frac{S_1^2(n1)}{n1} + \frac{S_1^2(n2)}{n2} \right]^{1/2}$$

2.5.- ANALYSE DES SORTIES

Si la vérification d'un modèle détermine la correction de la traduction du modèle au programme de simulation et si la validation détermine la précision de la représentation du système par le modèle, l'analyse des sorties du modèle détermine la représentativité et l'exactitude de ces derniers avant leur interprétation.

En effet, lorsque le modèle est stochastique, notre préoccupation durant cette analyse sera: "Comment déterminer de bonnes estimations de nos mesures de performances, en jouant sur les courants aléatoires et la durée de la simulation. Si par contre, le modèle est déterministe, le seul paramètre qui interviendra dans l'estimation des mesures de performances sera la durée de la simulation.

Pour l'étude que nous avons projeté sur notre système et les objectifs que nous nous sommes fixés, notre modèle sera déterministe. En effet, les temps de service sont fixes, ils correspondent aux temps d'exécution des procédures du système réel. Quant aux débits des voies basse vitesse et au nombre de connexions à établir, ils seront fixés selon la situation à étudier. Ceci dit,

notre système peut tout de même, être considéré sous une forme stochastique et son caractère stochastique sera alors déterminé par l'aléa:
— du nombre de connexions,
— de la durée entre les connexions ou inter-connexions, et
— du débit de l'information sur une connexion,
ces informations étant fournies par le terminal superviseur.

Il ne sera quand même pas exécuté sous sa forme stochastique, en raison de l'importance de la durée des inter-connexions qui peut s'élever aux 2 heures, et qui par conséquent, étend l'exécution du programme de simulation sur 25 jours.

Quant à l'analyse des sorties, elle ne nécessitera point d'estimation des paramètres de mesure de performance par intervalle de confiance. Une simple exécution en état de stabilité déterminera la moyenne de ces paramètres.

Cependant, envisageant la possibilité d'une exécution de notre modèle sous sa forme stochastique sur un système très rapide, nous présenterons, sans trop de détails, les différentes procédures d'estimation par intervalle de confiance.

2.5.1.- Etat stationnaire/ état transitoire

L'évolution dynamique du système peut être définie par trois états:
— l'état initial,
— l'état transitoire et
— l'état stationnaire, si celui-ci existe.

Partant de l'état initial, le système passe par une série d'états transitoires avant d'atteindre le régime stationnaire. Ces transitions sont dues aux effets des conditions initiales. Une fois l'état stationnaire atteint, l'évolution du système devient indépendante des conditions initiales.

2.5.2.- Simulation terminale/ simulation en état stationnaire [18]

Selon l'analyse à porter sur le système (ce que cherche à connaître l'analyste sur le système), deux types de simulation

peuvent être définis:
— la simulation terminale,
— la simulation en état stationnaire.

La simulation terminale est une simulation dans laquelle la mesure des performances du modèle est définie pour l'intervalle de temps $[0, T_E]$ où T_E est l'instant d'occurrence de l'évènement spécifique E. Comme la mesure des performances d'une simulation terminale dépend explicitement de l'état du système simulé au temps E, des précautions doivent être prises quant au choix des conditions initiales.

La simulation en état stationnaire est une simulation dans laquelle les mesures de performance sont définies sur une durée de la simulation tendant vers l'infini. Comme on ne dispose pas dans ce type de simulation, d'évènement naturel E pour terminer la simulation, la longueur de celle-ci est fixée de manière suffisamment grande pour donner une bonne estimation des quantités à évaluer. Souvent cette longueur est déterminée par des considérations de coût.

2.5.3.- Construction de l'intervalle de confiance [20]

La précision des sorties d'un modèle, sachant l'aléa des distributions probabilistes utilisées en entrée, constitue la base de toute analyse des sorties d'un modèle.

En effet, les résultats de simulations à l'aide de courants aléatoires différents (ou «Streams») ou de longueur différente, montrent bien la variabilité et l'aléa de ces sorties. Il est donc inconcevable de procéder à une seule exécution de la simulation, considérer ses résultats comme représentatifs et entamer leur interprétation. L'approche usuelle permettant la saisie de ces résultats est la construction d'un intervalle de confiance pour le paramètre $\bar{X}_I$, qui est la moyenne de l'échantillon. De nombreuses procédures ont été développées pour la construction de cet intervalle de confiance, et leur utilisation dépend souvent du type de simulation projeté (simulation terminale ou simulation en état stationnaire).

2.5.3.1.- Procédure des répliques

Elle propose des exécutions séparées, à l'aide de courants aléatoires différents, donc des exécutions indépendaotes de la simulation. A chaque exécution i, la moyenne de l'échantillon, X_i est calculée, et la variance $VAR[X_I]$ est estimée à l'aide de l'équation:

$$S^2_{\bar{X}} = \frac{1}{I} \frac{\sum_{i=1}^{I} (X_i - \bar{X}_I)}{I-1} \qquad \text{où} \qquad \bar{X}_I = \frac{\sum_{i=1}^{I} X_i}{I}$$

et l'intervalle de confiance estimé à $100(1-\alpha)$ est:

$$\bar{X}_I \pm t_{I-1,1-\alpha/2} \; S_{\bar{X}}$$

Cette procédure est utilisée aussi bien pour la simulation terminale que pour la simulation en état stationnaire.

2.5.3.2.- Procédure des «batch means»

Elle divise la durée d'une exécution en intervalles égaux puis calcule X_i comme une moyenne de l'intervalle i. les X_i seront supposés indépendants, quoique réellement, ils sont corrélés en raison de l'autocovariance qui existe entre les valeurs de la fin d'un intervalle et celles du début de l'intervalle suivant. Seulement, si l'on choisissait la longueur de ces intervalles suffisamment grande puis on les séparait par des périodes dites mortes [22], la corrélation entre les X_i serait approximativement nulle. De plus, si la longueur des intervalles est suffisamment grande, les X_i seront normalement distribués d'après le théorème central limite. Ainsi la variance sera estimée par:

$$S^2_{\bar{X}} = \frac{1}{I} \frac{\sum_{i=1}^{I} (X_i - \bar{X}_I)}{I-1} \qquad \text{où} \qquad \bar{X}_I = \frac{\sum_{i=1}^{I} X_i}{I}$$

et l'intervalle de confiance estimé à $100(1-\alpha)$ est:

$$\bar{X}_I \pm t_{I-1,1-\alpha/2} \; S_{\bar{X}}$$

Cette procédure est utilisée pour la simulation en état stationnaire. Elle présente l'avantage de ne poser le problème des conditions initiales qu'une seule fois.

2.5.3.3.- Procédure des cycles régénératifs

Cette procédure est similaire à celle des «batch means», en ce sens qu'elle divise la durée d'une exécution, en intervalles appelés cycles. Ces cycles ne sont plus de taille égale mais déterminés par l'occurrence d'un événement spécifique du système. Cette manière de définir les cycles assure l'indépendance des observations et élimine la covariance provoquée par l'utilisation des intervalles. La longueur du cycle n'est pas prédéterminée mais aléatoire. Suivant le développement de Crane et Lemoine [31], soit:

Y_i = la valeur totale du paramètre au cycle i,
L_i = la longueur du cycle i,
X_{ij} = la jième valeur du cycle i,
I = le nombre de cycles de la simulation.

$$Y_i = \sum_{j=1}^{L_i} X_{ij} \qquad (4.10)$$

$$\sum_{i=1}^{I} L_i = N \qquad (4.11)$$

La moyenne de tous les échantillons de l'exécution, $\bar{X}_I$ est:

$$\bar{X}_I = \sum_{i=1}^{I} \sum_{j=1}^{L_i} \frac{X_{ij}}{N} = \frac{\sum_{i=1}^{I} Y_i / I}{\sum_{i=1,I} L_i / I} = \bar{Y}_I / \bar{L}_I \qquad (4.12)$$

Une estimation de la variance de $\bar{X}_I$ est donnée par:

$$S^2_{\bar{X}} = \frac{S^2}{I\,\bar{L}_I^2} \qquad (4.13)$$

où:
$$S^2 = S_Y^2 - 2\,\bar{X}_I S_{YL} + \bar{X}_I^2 S_L^2 \qquad (4.14)$$

et
$$S_Y^2 = \frac{1}{I-1} \sum_{i=1}^{I} (Y_i - \bar{Y}_I)^2 \qquad (4.15)$$

$$S_L^2 = \frac{1}{I-1} \sum_{i=1}^{I} (L_i - \bar{L}_I)^2 \qquad (4.16)$$

$$S_{YL}^2 = \frac{1}{I-1} \sum_{i=1}^{I} (Y_i - \bar{Y}_I)(L_i - \bar{L}_I) \qquad (4.17)$$

Et l'intervalle de confiance approximé à $100(1-\alpha)$ est:

$$\bar{X}_I \pm t_{I-1,1-\alpha/2} \; S_{\bar{X}}$$

2.5.3.4.- Procédure de la covariance

Dans le cas où les observations sont corrélées, $VAR[\bar{X}_I]$ est égal à:

$$VAR[\bar{X}_I] = \frac{1}{I} \left[\frac{\sum_{i=1}^{I} (X_i - \bar{X}_I)^2}{I-1} + 2 \sum_{i=1}^{I-1} \left(1 - \frac{|h|}{I}\right) R_h \right] \qquad (4.18)$$

où:

$$R_h = cov[X_i, X_j] \quad \text{et} \quad h = j-i$$

Après estimation de l'autocovariance, R_h est obtenue et l'on peut donc calculer:

$$S_{\bar{X}}^2 = \frac{1}{I} \left[\frac{\sum_{i=1}^{I} (X_i - \bar{X}_I)^2}{I-1} + 2 \sum_{i=1}^{I-1} \left(1 - \frac{|h|}{I}\right) \hat{R}_h \right] \qquad (4.19)$$

où:

$$\hat{R}_h = \frac{1}{I} \sum_{i=1}^{I-h} (X_i - \bar{X}_I)(X_{i+h} - \bar{X}_I) \qquad (4.20)$$

Quant à l'intervalle de confiance approximé à $100(1-\alpha)$ est:

$$\bar{X}_I \pm t_{I-1,1-\alpha/2} \; S_{\bar{X}}$$

2.5.4.- Stratégie de départ

Les conditions initiales, établies pour une simulation, influencent de manière considérable la sortie du modèle. Lorsque la simulation est terminale, les résultats obtenus durant la phase transitoire, quoique non stables, sont très représentatifs de la sortie du système réel. Par contre, lorsque la simulation tend vers l'état stationnaire, les résultats de la phase transitoire peuvent influencer défavorablement, puisque peu représentatifs du régime stationnaire, la sortie finale du modèle.

La stratégie de départ consiste donc, à fixer les conditions initiales d'une simulation et à spécifier une procédure pour l'établissement du point de troncature d, à partir duquel les valeurs de l'échantillon seront incluses dans l'estimation en cours. Seulement, l'utilisation de cette stratégie doit se faire en conjonction avec les procédures d'estimation utilisées. Si l'estimation est obtenue à partir de la procédure des cycles régénératifs, la stratégie de départ devient très aisée; il suffira de démarrer l'exécution sur un état régénératif, de telle sorte que le premier cycle commence immédiatement et aucune troncature n'est requise. Si l'estimation est basée sur la procédure des «batch means», l'établissement des conditions initiales est alors effectué une seule fois. Cependant, si la procédure des répliques est utilisée, la stratégie de départ est répétée autant de fois que de répliques et d'énormes précautions doivent être prises quant à son établissement.

2.5.4.1.- Etablissement des conditions initiales

Trois règles de base sont proposées pour l'établissement des conditions initiales:

Règle 1 : démarrer le système vide et inactif

Elle présente l'avantage d'une implémentation facile, et fournit des états très représentatifs des modèles à petite échelle tels que les processus M/M/1.

Règle 2 : démarrer le système à un état stationnaire

Elle spécifie que les meilleures conditions initiales sont celles qui présentent la plus grande probabilité de se produire. Cela suppose donc, une simulation initiale pour déterminer ces

conditions.

Règle 3 : démarrer le système à un état stationnaire moyen.

Elle recommande l'utilisation de la moyenne des états attendus. Cette moyenne peut être obtenue à travers une étude analytique approximative du modèle.

2.5.4.2.- Procédures de troncature

La question de troncature invoque une double considération. La suppression des valeurs initiales tend à réduire le biais des estimateurs de sortie. Elle fait croître, cependant l'imprécision de l'estimateur de VAR[$\bar{X}_i$], du fait de réduire le nombre d'observations. Un compromis doit donc être réalisé entre la réduction du biais et la réduction de la variance. De nombreux auteurs ont tenté de fournir des règles pour détecter le moment favorable pour une troncature. Ces règles sont développées dans les ouvrages [32], [33], [34], [35] et [36], et évaluées et critiquées dans les ouvrages [37], [38] et [39]. Cette critique décommande la troncature sur les petits modèles.

2.5.5.- Analyse des sorties du modèle

Notre modèle étant déterministe, nous n'avons nullement besoin d'estimer ses paramètres de mesure de performance par intervalle de confiance. Ces derniers sont invariables au cours des différentes exécutions de même longueur. Par contre, une analyse sur la durée de la simulation s'impose. Pour cela, il suffit d'exécuter le modèle sur une longue durée (en état stationnaire) pour déterminer la moyenne de ses paramètres.

La stabilité du régime de notre modèle a pu être vérifiée après obtention, à partir de deux durées de simulation très différentes (1.1 min et 6 min), de résultats très proches. La comparaison de ces résultats a révélé un très faible pourcentage d'erreur (inférieur à 1.43 % pour la moyenne des files et inférieur à 0.81 % pour la longueur maximum de ces files). Ces résultats sont présentés dans le tableau 4.4. Le pourcentage d'erreur a été calculé comme suit:

$$\% \text{ d'erreur} = \frac{|\text{Long. sur 6 min d'exéc.} - \text{Long. sur 1.1 min d'exéc.}|}{\text{Long. sur 6 min d'exécution}}$$

Quant aux conditions d'exécution de ces deux simulations, elles sont les suivantes:

— les 8 périphériques sont connectés,
— la durée entre deux connexions est de 1 seconde,
— le débit de l'information est de 1200 bits/s,
— le système démarre vide,
— une troncature est effectuée après une minute d'exécution.

Tableau 4.4.

	Exécution sur 1.1 min		Exécution sur 6 min			
Numéro de la file	Longueur moyenne	Longueur maximum	Longueur moyenne	longueur maximum	Longueur moyenne % erreur	Longueur maximum % erreur
11	28.673	55	28.633	55	0.14 %	0.00 %
12	27.899	55	28.303	55	1.43 %	0.00 %
13	28.226	54	28.291	55	0.23 %	1.81 %
14	27.283	50	27.347	50	0.23 %	0.00 %
15	27.196	50	27.214	50	0.07 %	0.00 %
16	30.660	55	30.748	55	0.29 %	0.00 %
17	31.127	54	30.694	55	1.41 %	1.81 %
18	30.640	55	30.638	55	0.01 %	0.00 %
21	14.808	44	14.862	44	0.36 %	0.00 %
22	14.607	44	14.504	44	0.71 %	0.00 %
23	15.379	44	15.471	44	0.59 %	0.00 %
24	13.435	44	13.594	44	1.17 %	0.00 %
25	13.575	44	13.516	44	0.44 %	0.00 %
26	14.508	44	14.556	44	0.33 %	0.00 %
27	14.088	44	14.229	44	0.99 %	0.00 %
28	15.227	44	15.254	44	0.18 %	0.00 %

2.6.- INTERPRETATION DES RESULTATS ET EXPLOITATION DU MODELE

Comme nous l'avons déjà signalé, le but de cette simulation est de mesurer les performances de notre système en mode de fonctionnement normal, puis étudier son comportement en fonction de ses paramètres critiques, et sous les conditions de stress, afin de déterminer les grandeurs optimales des paramètres qui nous intéressent et de s'armer des mesures qui s'imposent pour parer à la moindre de ses défaillances (blocage, débordement, ...).

L'une de nos préoccupations principales, lors de cette étude, est la grandeur de la taille des mémoires tampons, lorsque le débit maximum cumulé des voies basse vitesse est égal au débit de la voie composite (taux de concentration de 100%). Or, cette grandeur est fonction de deux paramètres essentiels, qui sont:

— la longueur de la trame, et
— une constitution simultanée d'une trame sur les huit (8) mémoires tampons de réception.

Nous étudierons chacun de ces deux paramètres, afin de déterminer la plus petite taille mémoire tampon de réception qui ne pose aucun problème de débordement dans les conditions de stress.

Pour des taux de concentration supérieurs à 100%, le problème est tout autre. Plus la taille des mémoires tampons est grande, plus la concentration pourra être élevée, sans pour autant trop interrompre le débit des voies basse vitesse lorsqu'elles sont toutes actives en même temps. Mais quelles précautions faudra-t-il prendre alors?

2.6.1.- Optimisation de la longueur de la trame

Il est clair que plus la longueur de la trame est petite, plus la taille des mémoires tampons sera petite, et la charge de la voie haute vitesse mieux exploitée, car une trop grande longueur de la trame peut immobiliser assez longtemps, la transmission sur la voie haute vitesse, le temps d'attente de la constitution d'une trame étant très long. Cependant, sachant que deux caractères de contrôle sont rajoutés à chaque trame, et donc transmis sur la voie haute vitesse, une trop petite longueur de la trame risque de charger inutilement (à l'aide des caractères de contrôle) la voie haute

vitesse. Il s'agit donc, de déterminer la plus petite longueur de la trame, qui minimise les silences sur la voie haute vitesse, sans pour autant les combler par l'envoi des caractères de contrôle.

Les résultats obtenus à la suite de simulations avec des longueurs de trame différentes, sont présentés dans le tableau 4.5. Les conditions de la simulation sont:

— nombre de voies fonctionnelles = 8,
— intervalle de temps séparant le lancement des 8 voies = 1 sec.,
— troncature après 7 secondes de fonctionnement,
— durée de la simulation = 3 min.,
— temps d'exécution de la simulation = 2 heures.

Tableau 4.5.

Long. de la trame	Taille de la file	Taille des tampons de réception	Taille des tampons d'émission	Données émises	Charge de la voie HV
6	237	180-180-181-182 182-183-184-183	11-10-11-11 11-10-10-11	20240	73.36 %
7	89	85-85-85-85 85-84-84-85	7-7-7-7 7-7-7-7	21280	77.13 %
8	9	16-16-16-16 16-17-16-17	15-13-14-13 14-15-14-14	22055	79.93 %
9	3	11-12-11-11 12-11-12-12	9-10-9-9 9-9-9-9	22066	79.97 %
10	7	19-18-18-18 18-18-18-19	17-16-16-17 17-15-16-17	22066	79.97 %
11	2	14-13-13-13 12-12-11-11	10-10-10-10 10-10-10-10	22076	80.01 %
12	7	22-22-21-22 21-21-21-22	20-18-18-14 20-18-20-21	22071	79.99 %
13	1	14-14-14-14 13-13-13-13	12-12-12-12 12-12-12-12	22063	79.96 %
14	1	16-14-16-14 15-14-14-16	13-13-13-13 13-12-13-13	22067	79.97 %

Long. de la trame	Taille de la file	Taille des tampons de réception	Taille des tampons d'émission	Données émises	Charge de la voie HV
15	7	27-27-27-27 27-27-27-27	23-23-23-18 17-21-24-25	22064	79.96 %
16	3	21-21-21-21 21-21-21-21	15-15-15-15 15-15-16-16	22062	79.96 %
17	4	17-18-19-20 21-22-22-23	16-16-15-15 15-15-16-16	22071	79.97 %
18	2	22-22-18-22 22-21-22-22	17-16-16-17 16-16-16-16	22066	79.97 %
19	2	21-22-20-20 21-19-19-20	17-17-17-17 17-17-17-17	22068	79.98 %
20	7	35-35-35-35 35-35-35-35	30-26-30-30 30-30-30-31	22055	79.93 %
21	1	23-21-21-23 21-21-22-23	19-19-19-19 19-19-19-19	22069	79.98 %
22	1	23-24-23-23 23-23-22-22	20-20-20-20 20-20-20-20	22066	79.97 %
23	1	24-23-23-24 23-24-24-24	21-20-21-20 21-21-20-21	22053	79.92 %
24	7	42-42-42-42 42-42-42-42	35-25-24-22 22-22-34-36	22160	80.32 %
25	2	28-28-27-25 26-29-28-28	22-23-22-23 23-22-22-22	22065	79.97 %
26	1	26-26-26-26 26-27-27-27	23-24-24-23 23-23-23-23	22052	79.92 %
27	1	28-27-27-27 27-27-27-27	24-24-24-24 24-24-24-24	22060	79.95 %
28	1	31-29-28-28 30-28-28-31	25-25-25-25 25-25-24-25	22065	79.97 %
29	1	31-30-29-29 29-29-29-29	26-26-26-26 26-26-25-25	22059	79.94 %
30	7	52-52-52-52 52-52-52-53	39-38-34-35 36-44-44-44	22047	79.90 %
31	1	31-31-31-31 31-31-31-32	28-27-28-28 28-28-27-28	22049	79.91 %

Long. de la trame	Taille de la file	Taille des tampons de réception	Taille des tampons d'émission	Données émises	Charge de la voie HV
32	1	35-35-35-35 36-36-36-36	28-28-28-28 28-28-28-28	22044	79.89 %
33	1	34-34-33-33 33-33-33-33	29-29-29-29 29-29-29-29	22067	79.97 %
34	2	34-34-36-36 37-37-39-39	30-30-30-30 31-30-30-30	22054	79.93 %
35	1	39-35-35-36 35-37-35-38	31-31-31-31 31-31-31-30	22051	79.92 %
36	2	43-44-39-44 44-40-44-44	32-32-32-32 32-32-32-32	22030	79.84 %
37	1	40-40-40-40 37-37-37-37	33-33-33-33 33-33-33-32	22046	79.90 %
38	2	40-42-38-38 38-38-38-40	33-34-34-33 33-34-34-33	22052	79.92 %
39	2	39-40-42-42 44-45-46-47	35-34-35-34 34-34-35-35	22024	79.82 %
40	7	69-69-69-69 69-69-69-69	55-49-42-40 41-45-52-56	22036	79.86 %
45	2	54-54-49-54 54-50-54-54	40-40-39-40 39-40-40-40	22046	79.90 %
50	1	56-55-55-51 51-56-55-55	44-44-44-44 44-44-44-44	22042	79.88 %
60	7	91-97-103-103 103-103-103-104	64-63-61-66 64-74-82-82	22015	79.79 %
70	1	77-70-70-74 70-70-72-77	61-62-62-61 61-62-62-61	22050	79.91 %
80	3	104-104-104-104 105-104-105-105	70-70-70-70 70-70-70-70	21999	79.73 %
90	2	108-108-99-108 108-99-109-108	78-79-79-78 79-79-78-79	21958	79.58 %
100	2	110-111-111-102 100-110-111-111	87-87-88-88 88-87-87-88	21905	79.39 %
110	1	110-111-113-114 115-116-118-119	95-95-95-95 96-97-97-97	21986	79.68 %

Long. de la trame	Taille de la file	Taille des tampons de réception	Taille des tampons d'émission	Données émises	Charge de la voie HV
120	7	205-205-193-205 205-193-205-205	156-136-107-106 110-126-123-120	21956	79.58 %
130	2	153-149-146-143 139-137-133-130	115-115-114-113 112-112-113-113	22008	79.76 %
140	1	157-140-140-140 140-142-148-154	121-122-122-123 123-122-122-122	21978	79.66 %
150	2	166-151-165-150 151-166-166-165	131-129-130-131 132-131-130-130	22040	79.88 %
160	1	176-175-176-176 176-176-176-176	140-140-140-139 139-139-139-139	21918	79.44 %
170	1	177-170-170-175 170-170-172-170	148-147-149-148 147-149-148-148	21987	79.67 %
180	2	216-216-198-216 216-198-216-216	156-156-157-157 156-157-157-156	21957	79.58 %
190	0	190-190-190-190 190-190-190-190	166-165-165-165 165-165-165-165	21954	79.57 %
200	2	220-220-221-201 201-220-220-221	173-175-174-175 174-173-175-174	21802	79.02 %
210	1	231-210-223-210 214-210-210-231	182-184-183-183 183-182-184-182	21925	79.46 %

$$\text{Charge de la voie HV} = \frac{\text{Nb. d'émissions} - 2(\text{Nb. de trames émises})}{\text{Nb. d'interruptions d'émission}}$$

Nb. d'interrupt. d'émission = Nb. d'émissions + Nb. de silences

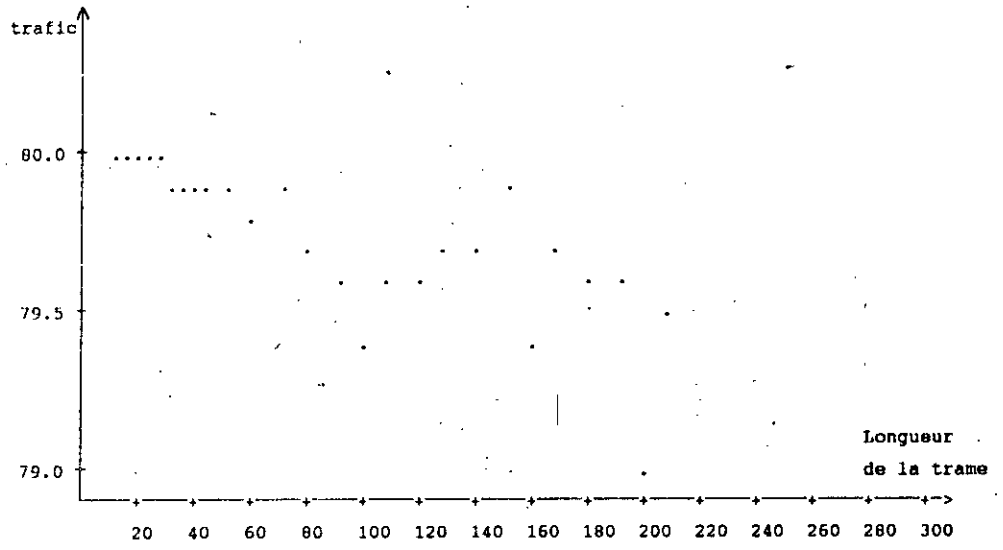


Figure 4.5. Trafic sur la voie composite en fonction de la longueur de la trame

Ces résultats révèlent que le trafic sur la voie composite est de manière générale, très stable et élevé (≈ 80 %) pour une longueur de la trame comprise entre 8 et 70 caractères. Il atteint son point culminant (80.32 %) pour une longueur de trame égale à 24 caractères. Par contre, la taille des tampons de réception et d'émission, pour cette même plage de longueurs, fluctue quant à elle, selon les longueurs de la trame et de la file d'attente. Elle croît, pour une longueur donnée de la file d'attente, dans le même sens que la longueur de la trame (voir les figures 4.6, 4.7, 4.8, 4.9, 4.10, 4.11). En effet, plus les trames sont grandes et nombreuses, plus l'espace requis pour leur sauvegarde sera grand.

Pour une longueur de la trame inférieure à 6 ou supérieure à 200 caractères, la taille mémoire requise pour les tampons de réception, d'émission et la file d'attente, est considérable et l'exécution de la simulation est interrompue pour manque d'espace mémoire. Lorsque la trame dépasse les 60 caractères, la taille des tampons de réception et d'émission est également très élevée et le trafic sur la voie composite est diminué.

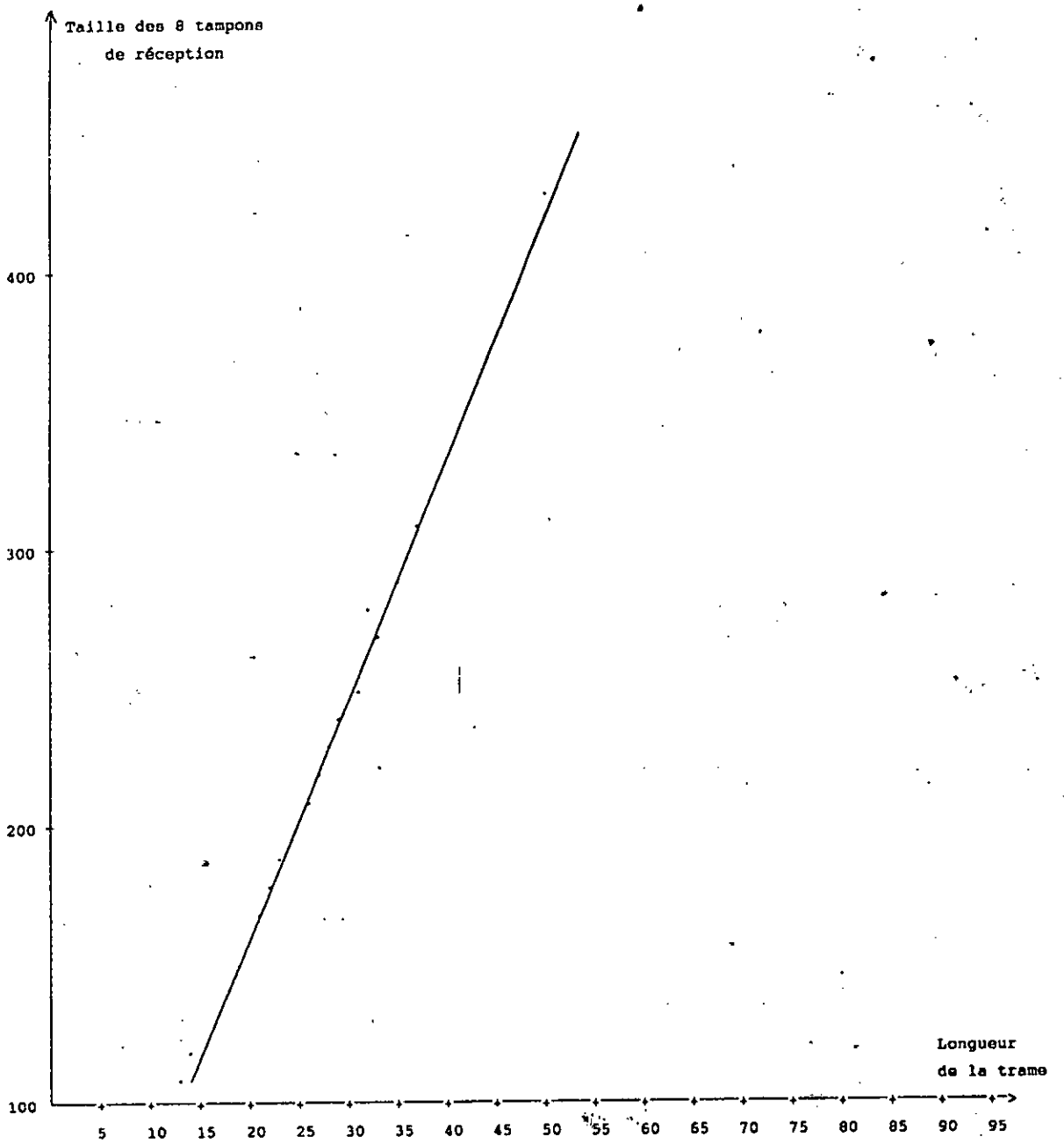


Figure 4.6. Taille des 8 tampons de réception en fonction de la longueur de la trame et pour une longueur de la file d'attente égale à 1

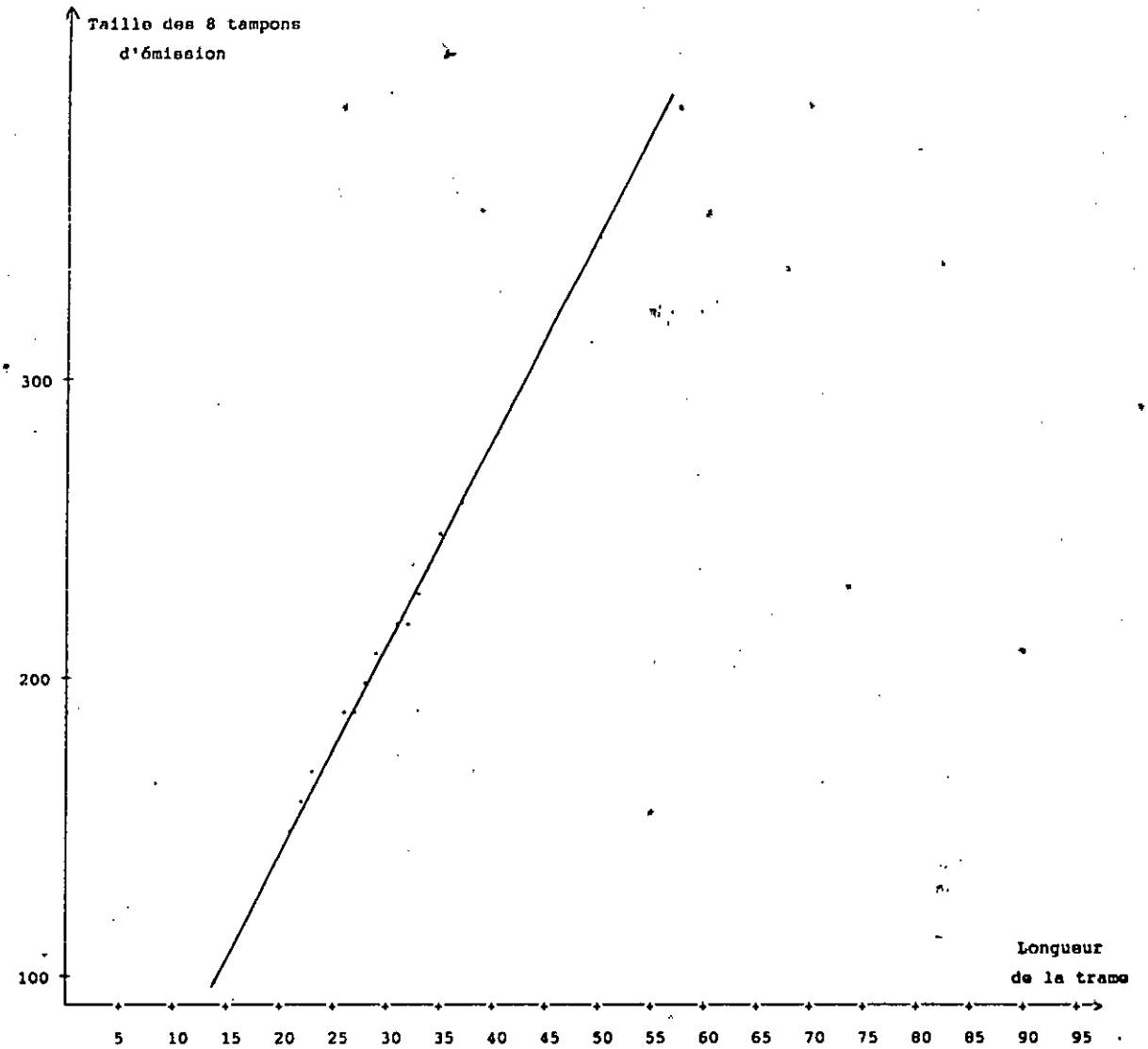


Figure 4.7. Taille des 8 tampons d'émission en fonction de la longueur de la trame et pour une longueur de la file d'attente égale à 1

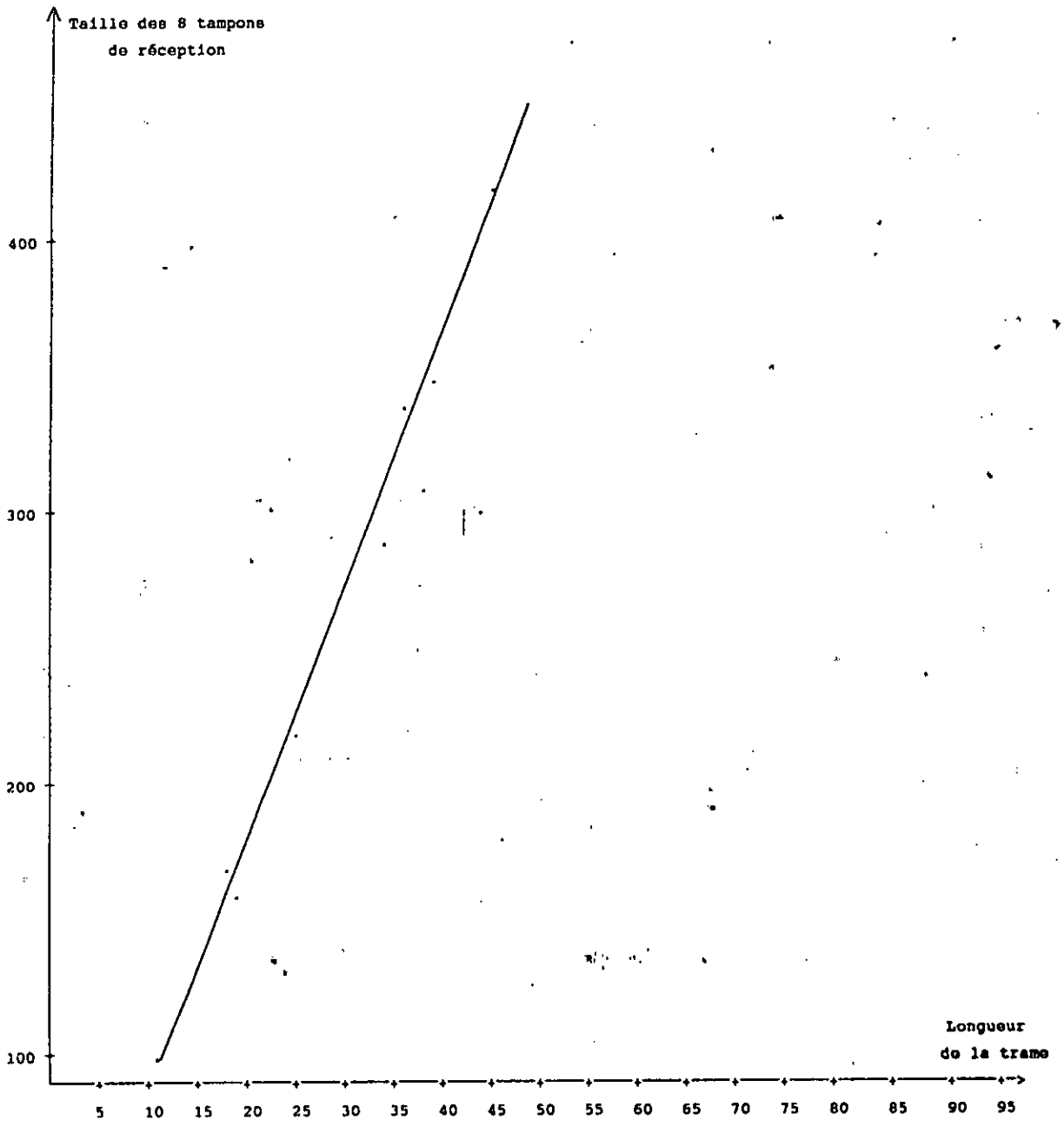


Figure 4.8. Taille des 8 tampons de réception en fonction de la longueur de la trame et pour une longueur de la file d'attente égale à 2

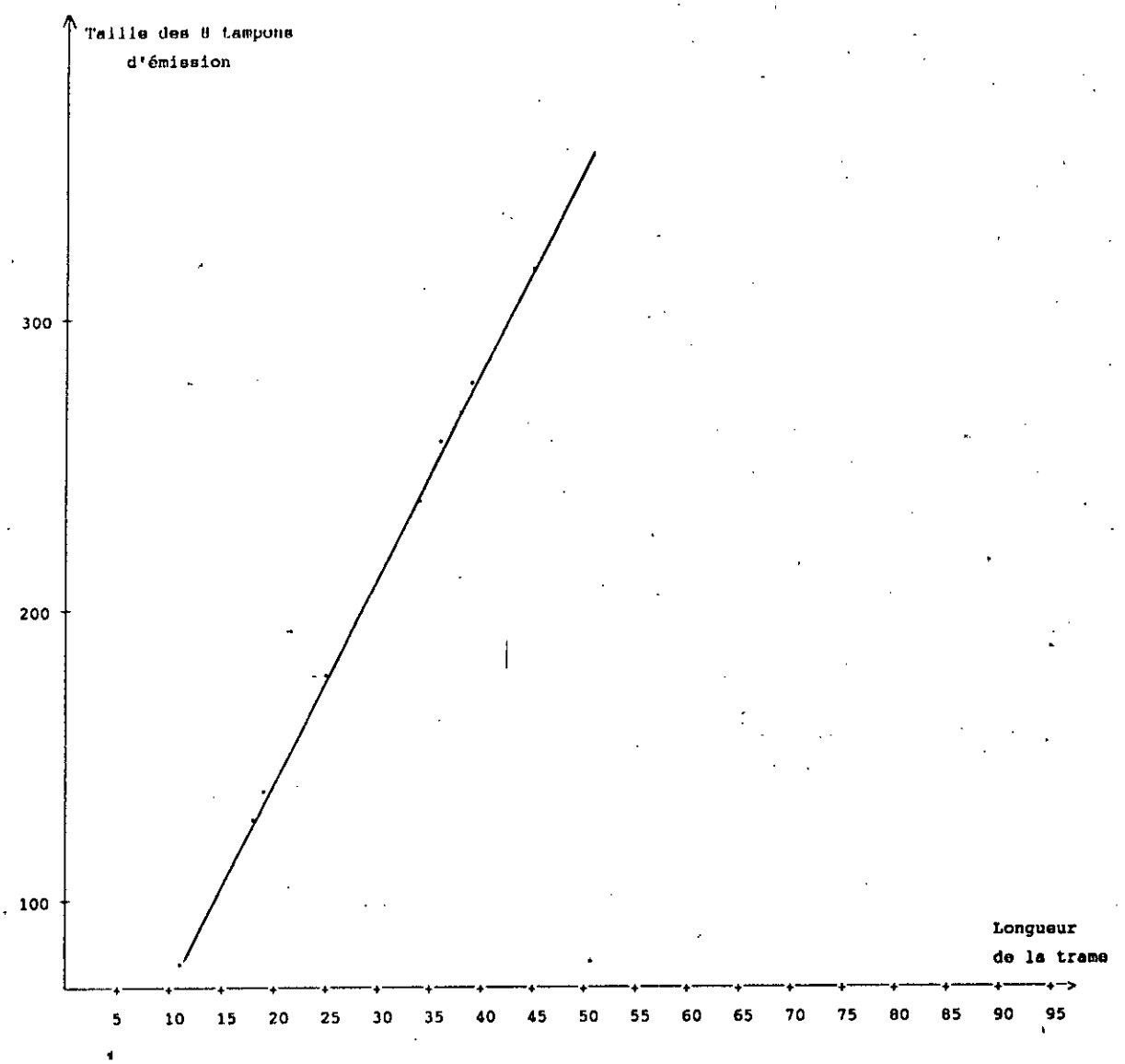


Figure 4.9. Taille des 8 tampons d'émission en fonction de la longueur de la trame et pour une longueur de la file d'attente égale à 2

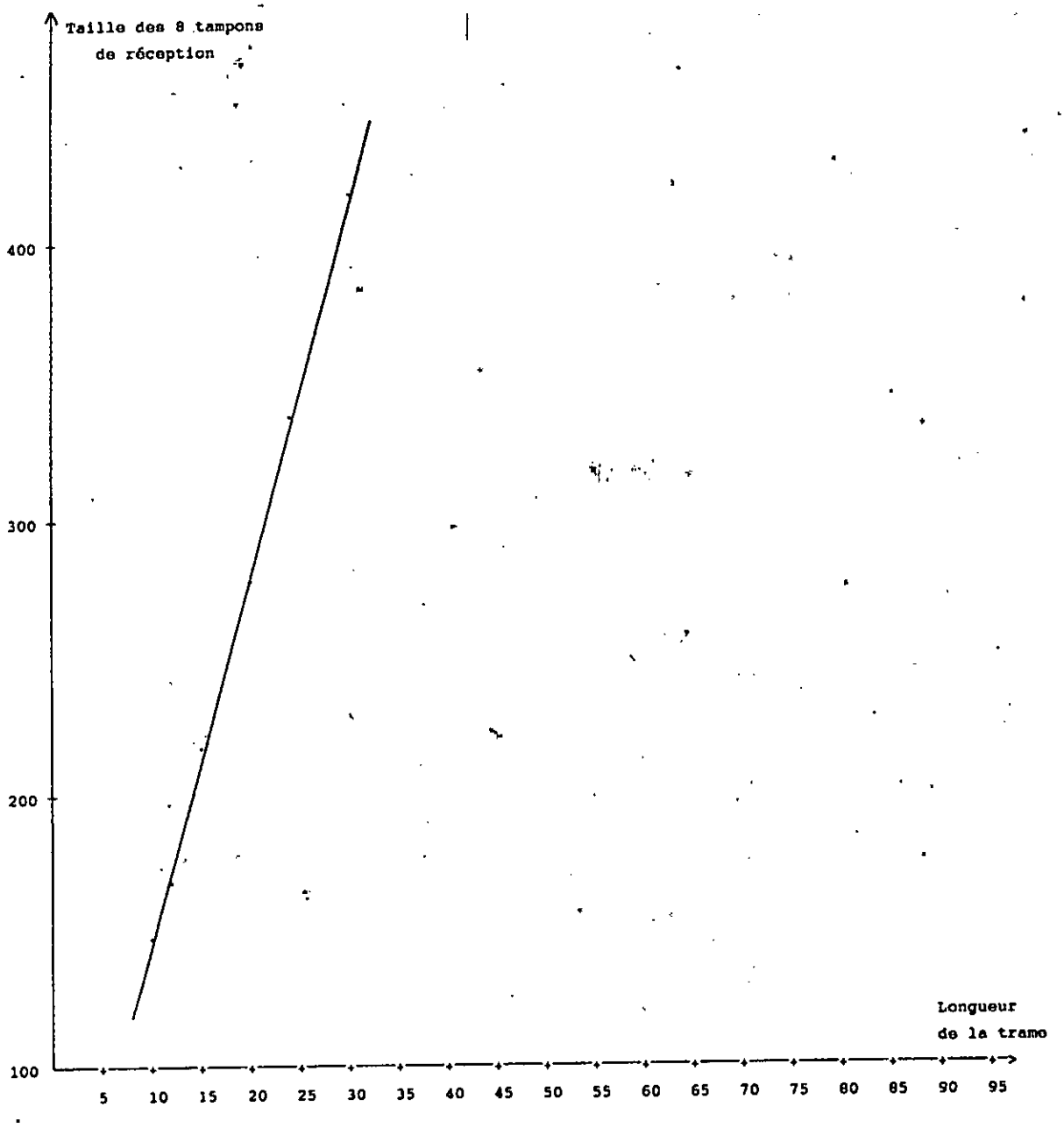


Figure 4.10. Taille des 8 tampons de réception en fonction de la longueur de la trame et pour une longueur de la file d'attente égale à 7

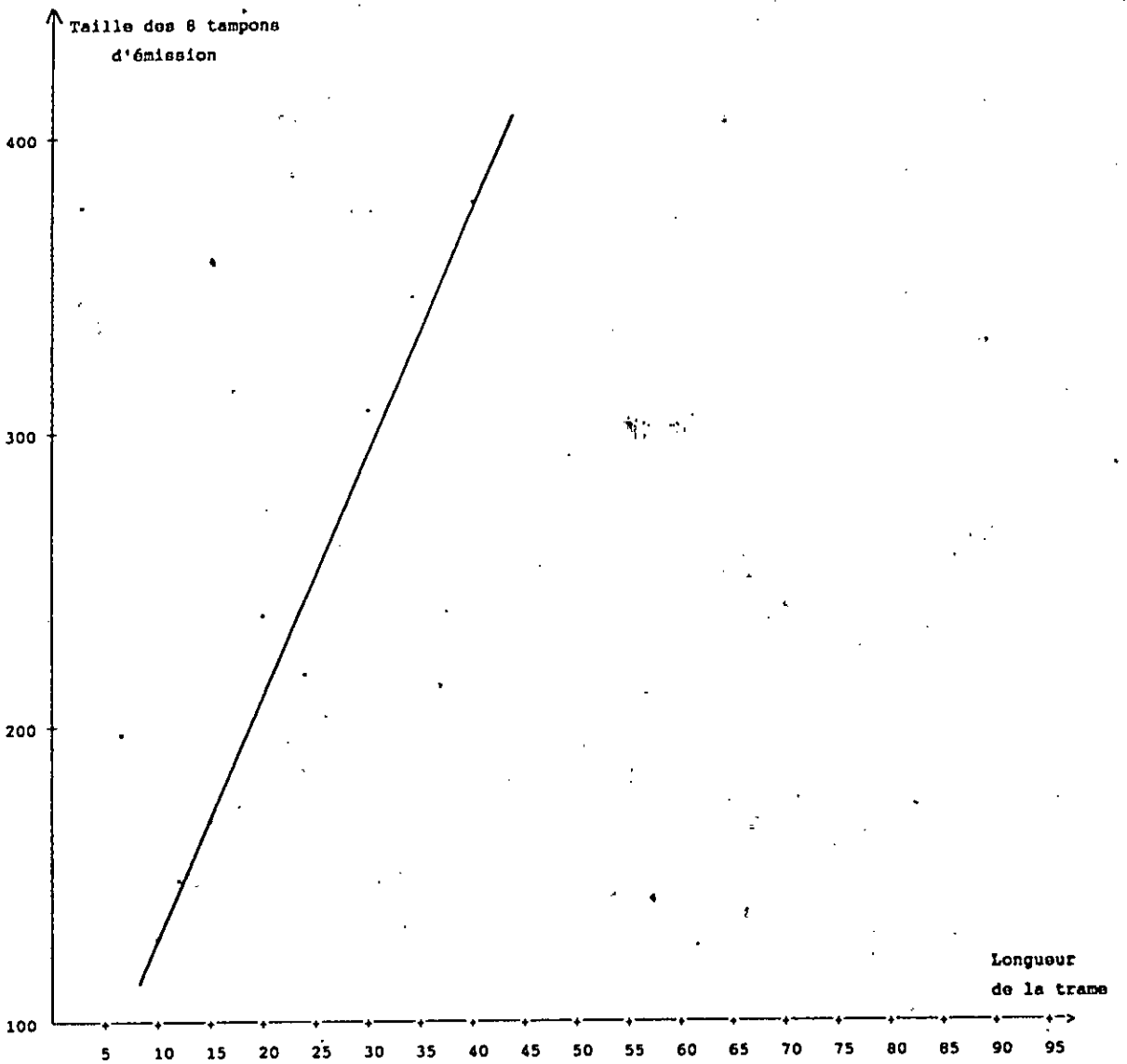


Figure 4.11. Taille des 8 tampons d'émission en fonction de la longueur de la trame et pour une longueur de la file d'attente égale à 7

2.6.2.- Constitution simultanée d'une trame sur les huit mémoires tampons de réception

Lorsque les huit voies basse vitesse sont lancées simultanément, qu'elles pédalent à la même fréquence et que la longueur de leurs trames est identique, les constitution et enfilement d'une trame prête pour l'émission ont toutes les chances de se produire en même temps, pour les huit voies. Ceci se répercute sur la taille de la file d'attente qui se trouve augmentée, et par conséquent, sur celles des tampons de réception et d'émission.

Seulement, démarrer les huit voies simultanément est une chose difficilement réalisable sur un système réel. Il peut, par contre, se produire le cas où la fréquence de lancement des différentes voies coïncide avec celle d'un démarrage simultané et produit la situation d'une constitution simultanée de trame sur les huit voies.

Pour illustrer ceci, nous avons effectué plusieurs simulations où le lancement des huit voies est simultané. Nous nous sommes inspirés pour le choix de nos tests des résultats du tableau 4.5. Nous avons tout d'abord misé sur les longueurs de trame fournissant un trafic optimal, puis sur des valeurs spécifiques de la longueur de la file d'attente. Les résultats obtenus sont représentés dans le tableau 4.6. Les conditions de la simulation sont:

— nombre de voies fonctionnelles = 8;
— lancement des 8 voies simultané,
— troncature après 7 secondes de fonctionnement,
— durée de la simulation = 3 min.,
— temps d'exécution de la simulation = 2 heures.

Ces résultats montrent bien l'attente simultanée d'une trame de chaque voie dans la file d'attente (longueur de la file égale à 7 trames). Comparativement aux résultats du tableau 4.5, nous constatons que la taille des tampons de réception et d'émission reste inchangée pour les situations où une constitution simultanée de trames s'est produite lors de la sortie des résultats du paragraphe 2.6.1 (longueur de la file d'attente égale à 7). Cette taille est, par contre, beaucoup augmentée, lorsque les résultats du tableau 4.5 fournissent une longueur de la file d'attente inférieure à 7 (1, 2 ou 4). Dans tous les cas, nous remarquons que la charge du trafic sur la voie composite est plus importante.

Tableau 4.6.

Long. de la trame	Taille de la file	Taille des tampons de réception	Taille des tampons d'émission	Données émises	Charge de la voie HV
10	7	19-18-18-18 18-18-18-19	17-16-15-17 17-14-16-17	22074	80.00 %
11	7	20-20-20-20 20-20-20-20	18-17-15-13 15-17-17-19	22077	80.01 %
12	7	22-22-22-22 22-21-21-22	20-17-16-16 15-17-19-20	22080	80.02 %
14	7	25-25-25-25 25-25-25-25	22-20-20-15 17-18-21-13	22071	79.99 %
17	7	30-30-30-30 30-30-29-31	26-23-21-19 19-22-24-25	22092	80.07 %
22	7	38-38-38-38 39-38-38-39	32-32-25-22 23-30-31-34	22097	80.08 %
24	7	42-42-42-42 42-42-41-42	35-24-25-24 35-31-30-36	22076	80.01 %

2.6.3.- Plus grand débit des voies basse vitesse égal à celui de la voie composite

Nous avons vu dans le paragraphe 2.1.2.4 que lorsque le débit d'une voie basse vitesse est égal à celui de la voie composite, une seule voie peut être fonctionnelle; et son rythme de fonctionnement est tel que l'on risque de ne pas disposer de suffisamment de temps pour boucler la boucle et revenir à temps pour la desservir, ce qui entraîne un débordement de la FIFO de réception.

Pour vérifier ceci, nous avons effectué une simulation sous les conditions précitées, à savoir:

— 1 seule voie fonctionnelle à 9600 bits/seconde,
— longueur de la trame = 22 caractères,
— durée de la simulation = 2 min.

Les résultats de cette simulation ne révèlent aucune situation de débordement de la FIFO de réception. La taille du tampon de réception requise est de 25 caractères. Celle du tampon d'émission est de 3 caractères.

2.6.4.- Plus grand débit des voies basse vitesse égal à 4800 bauds

Il est apparu dans le paragraphe 2.1.2.4 que lorsque le débit le plus élevé des voies basse vitesse est de 4800 bauds, et que les 8 voies sont fonctionnelles, un débordement de la FIFO de réception de la voie configurée à 4800 bauds risque de se produire. Ceci est du, d'une part à l'intensité du débit sur cette voie, et d'autre part au temps écoulé dans le service des autres voies.

Pour s'assurer de ceci, nous avons effectué une simulation sous les conditions suivantes:

— nombre de voies fonctionnelles = 8,
— débit des 8 voies égal respectivement à 4800, 1200, 600, 600, 600, 600, 600, 600,
— longueur de la trame = 22 caractères,
— durée de la simulation = 2 min.

Il en est ressorti, en effet, un débordement de la FIFO de réception de la voie configurée à une vitesse de 4800 bauds. Comme ce problème de débordement est résolu sur le système réel par un arrêt automatique du débit de l'information, en cas de remplissage de la FIFO, nous avons simulé cet arrêt par un renvoi de l'information qui déborde, dans une file auxiliaire. Ceci a résolu le problème de débordement, mais a révélé une surcharge du tampon d'émission relatif à la voie en question (1481 caractères), problème qui a d'ailleurs interrompu l'exécution de la simulation pour manque d'espace mémoire. La résolution de cette surcharge du tampon s'est effectuée à l'aide d'un vidage plus fréquent de ce dernier.

Les résultats obtenus sont alors les suivants:

S L A M I I S U M M A R Y R E P O R T

SIMULATION PROJECT MUX BY TEBIBEL

DATE 14/ 4/1991 RUN NUMBER 1 OF 1

CURRENT TIME .3000E+08
STATISTICAL ARRAYS CLEARED AT TIME .0000E+00

STATISTICS FOR TIME-PERSISTENT VARIABLES

	MEAN VALUE	STANDARD DEVIATION	MINIMUM VALUE	MAXIMUM VALUE	TIME INTERVAL	CURRENT VALUE
TRAME COURANTE	10.897	5.158	.00	18.00	*******	.00
NB CAR A EMETTRE	8.314	5.985	.00	17.00	*******	.00
NB TRAM. VOIE 1	1.179	.905	.00	3.00	*******	.00
NB TRAM. VOIE 2	.502	.500	.00	1.00	*******	.00
NB TRAM. VOIE 3	.107	.309	.00	1.00	*******	.00
NB TRAM. VOIE 4	.152	.359	.00	1.00	*******	.00
NB TRAM. VOIE 5	.179	.383	.00	1.00	*******	.00
NB TRAM. VOIE 6	.215	.411	.00	1.00	*******	.00
NB TRAM. VOIE 7	.242	.428	.00	1.00	*******	.00
NB TRAM. VOIE 8	.285	.451	.00	1.00	*******	.00
NB ARRIV. VOIE 1	7199.769	4156.751	.00	14399.00	*******	14399.00
NB ARRIV. VOIE 2	1800.454	1039.220	.00	3600.00	*******	3600.00
NB ARRIV. VOIE 3	900.511	519.618	.00	1801.00	*******	1801.00
NB ARRIV. VOIE 4	900.511	519.618	.00	1801.00	*******	1801.00
NB ARRIV. VOIE 5	900.511	519.618	.00	1801.00	*******	1801.00
NB ARRIV. VOIE 6	900.511	519.618	.00	1801.00	*******	1801.00
NB ARRIV. VOIE 7	900.511	519.618	.00	1801.00	*******	1801.00
NB ARRIV. VOIE 8	900.511	519.618	.00	1801.00	*******	1801.00

FILE STATISTICS

FILE NUMBER	ASSOC NODE LABEL/TYPE	AVERAGE LENGTH	STANDARD DEVIATION	MAXIMUM LENGTH	CURRENT LENGTH	AVERAGE WAIT TIME
1	AWAIT	1.661	.939	3	1	3833.208
2	AWAIT	.323	.468	1	0	2692.824
3	AWAIT	.176	.381	1	1	2930.885
4	AWAIT	.189	.392	1	1	3149.201

5		AWAIT	.196	.397	1	1	3271.998
6		AWAIT	.203	.402	1	1	3377.167
7		AWAIT	.212	.409	1	1	3528.576
8		AWAIT	.220	.415	1	1	3671.400
9			.000	.000	0	0	.000
10		AWAIT	.397	.489	1	0	74.157
11			26.078	14.054	62	13	60175.770
12			16.045	6.124	29	13	*********
13			9.554	4.904	22	15	*********
14			10.322	4.868	22	15	*********
15			10.763	4.816	22	15	*********
16			11.367	4.817	22	15	*********
17			11.815	4.877	22	15	*********
18			12.534	4.941	22	15	*********
19			2.015	2.049	7	0	47081.260
20			2.399	1.233	4	0	2637.784
21	I23	AWAIT	9.461	9.864	38	5	21854.040
22	I23	AWAIT	1.207	2.635	13	0	10091.200
23	I23	AWAIT	.567	1.873	13	0	9525.848
24	I23	AWAIT	.571	1.883	13	0	9591.152
25	I23	AWAIT	.567	1.871	13	0	9522.290
26	I23	AWAIT	.562	1.850	12	0	9442.244
27	I23	AWAIT	.568	1.859	11	0	9550.191
28	I23	AWAIT	.572	1.857	11	0	9609.211
29		PREEMPT	.000	.000	1	0	.000
30		PREEMPT	.254	.435	1	0	211.373
31	CR1	AWAIT	696.540	403.146	1397	1397	*********
32		CALENDAR	11.000	.000	13	11	226.142

REGULAR ACTIVITY STATISTICS

ACTIVITY INDEX/LABEL	AVERAGE UTILIZATION	STANDARD DEVIATION	MAXIMUM UTIL	CURRENT UTIL	ENTITY COUNT
1	.1669	.3729	1	1	17936
2	.1297	.3360	1	0	32436
3	.0128	.1124	1	0	1605
4	.1038	.3050	1	0	51915
5	.2338	.4232	1	0	30495
6	.0198	.1393	1	0	5496
7	.1433	.3504	1	0	30495
8	.0000	.0000	1	0	5496
9	.0130	.1134	1	0	2182
10	.0688	.2530	1	0	42972
11	.0372	.1893	1	0	3155
12	.0708	.2565	1	0	8366

2.6.5.- Taux de concentration supérieur à 100 %

Les résultats présentés jusqu'à lors supposent un taux de concentration de 100 % et un fonctionnement des voies basse vitesse à plein débit, ce qui donne une charge de la voie composite de l'ordre de 80 %. Or dans la réalité, ces voies ne fonctionnent jamais de manière ininterrompue, et la charge de la voie composite qui en découle se trouve réduite à l'ordre des 30 %. Pour prévenir ces longs silences de la voie composite, on augmente le débit cumulé des voies basse vitesse jusqu'à ce qu'il atteigne deux à quatre fois celui de la voie composite. Le cas d'un fonctionnement simultané de toutes ces voies basse vitesse à haut débit doit toutefois, être prévu et solutionné par un arrêt du débit de la voie dont le tampon est saturé.

Pour tester tout ceci, nous avons simulé le système décrit ci-dessus, avec un taux de concentration de 200 %. Plusieurs scénarios ont été élaborés pour traduire la durée des silences des voies basse vitesse.

SCENARIO 1

Ce scénario correspond à un taux de concentration de 100%.

SCENARIO 2

Ce scénario correspond à un taux de concentration de 200% et un fonctionnement simultané et à plein débit des huit voies basse vitesse.

SCENARIO 3

Ce scénario correspond à un taux de concentration de 200%. Les voies basse vitesse ne sont pas toutes fonctionnelles en même temps, mais leur temps de silence est très réduit, notamment pour les premières voies:

— la voie 1 démarre à 1.0 μs,
— la voie 2 démarre à 2.0 E+1,
— la voie 3 démarre à 2.0 E+2,
— la voie 4 démarre à 2.0 E+3,
— la voie 5 démarre à 2.0 E+4,
— la voie 6 démarre à 2.0 E+5,
— la voie 7 démarre à 2.0 E+6,
— la voie 8 démarre à 2.0 E+7.

SCENARIO 4

Ce scénario correspond à un taux de concentration de 200%. Les voies basse vitesse ne sont pas toutes fonctionnelles en même temps et leur temps de silence est plus élevé que celui du scénario 3; le démarrage des voies est espacé par une période de 25.0 E+5 μs.

Pour ces quatre scénarios, la longueur de la trame est de 22 caractères et le temps de simulation est de 2.0 E+7 μs. Pour les trois derniers scénarios, le débit cumulé des voies basse vitesse est de 19200 bauds à raison d'une vitesse de 2400 bauds par voie.

Tableau 4.7.

Scénario	Arrivées sur les voies BV	Débordement des voies BV	Taille des tampons de réception	Taille des tampons d'émission	Charge de la voie HV
1	2401-2281 2161-2041 1920-1801 1681-1561	0000-0000 0000-0000 0000-0000 0000-0000	23-23-23 23-23-23 22-22	20-20-20 20-20-20 20-20	79.88 %
2	4800-4800 4800-4800 4800-4800 4800-4800	4589-4588 4611-4610 4610-4588 4609-4588	101-102 101-102 102-102 103-102	42-38-35 29-25-35 29-38	07.02 %
3	4800-4800 4800-4800 4796-4752 4320-2400	4390-4412 4433-4435 4453-4452 0000-0000	102-102 103-101 101-102 23-23	93-69-69 75-73-59 18-18	36.63 %
4	4800-4200 3600-3000 2400-1801 1201-0601	1618-1635 1628-1622 1639-1634 0000-0000	102-101 102-102 101-101 23-23	307-306 306-305 192-032 018-018	48.85 %

La première conclusion à tirer de ces résultats est qu'il est absolument inutile d'augmenter le taux de concentration lorsque les voies basse vitesse fonctionnent en permanence à plein débit. Ceci joue même en la défaveur de la charge de la voie composite qui se trouve alors fortement diminuée (scénario 2 en particulier). Nous remarquons sur les scénarios 2, 3 et 4 que plus les silences sont importants, plus la charge de la voie composite est augmentée et le débordement des voies basse vitesse est diminué. En fait, pour maintenir une charge élevée de la voie composite lorsque le taux de concentration est supérieur à 100% et que les silences sont réduits, l'on doit disposer de grandes tailles de mémoire tampon.

Dans notre cas, nous n'avons pu mener notre simulation sur une taille du tampon de réception global supérieure à 800 octets. Au delà de cette limite, le programme de simulation s'interrompt pour manque d'espace mémoire.

2.7.- Apport de la simulation pour la conception pratique

La simulation de notre produit fut d'un grand apport pour le système réel conçu. En effet, elle nous a permis, à maintes reprises, de résoudre des situations de blocage très complexes, tantôt en déterminant la taille des mémoires tampons et de la file d'attente des trames, tantôt en calculant les fréquences de service qui correspondent dans notre système au temps d'exécution des programmes. Elle nous a également permis de détecter les conditions de stress, les tester et leur prévoir des solutions adéquates.

Outre cette contribution à l'élaboration et la mise au point du logiciel, la simulation du produit a permis une optimisation du système réel, notamment sur la taille des mémoires tampons, la longueur de la file d'attente et la longueur de la trame. L'optimisation de la taille des tampons et de la longueur de la file d'attente s'est traduite par une minimisation de l'espace mémoire utilisé. Quant à l'optimisation de la longueur de la trame, elle s'est faite dans le but de maximiser le rendement de la voie composite, et par conséquent augmenter l'efficacité de la ligne.

De plus, cette simulation met l'accent sur la nécessité de s'assurer de l'élévation des silences avant de prendre l'initiative d'augmenter les taux de concentration; contrairement à ce que l'on pourrait espérer, le rendement de la voie composite risque de chuter de manière très sensible avec l'augmentation du taux de concentration, si les silences sur la voie composite ne sont pas importants et la taille des tampons est assez réduite. A cet effet, il serait intéressant d'envisager une variabilité automatique des vitesses de transmission en fonction des silences de la voie composite: les vitesses augmenteraient lorsque les silences sont importants et diminueraient lorsqu'ils le sont moins. Ceci doit se faire non sans l'accomplissement d'une étude statistique bien menée.

Enfin, le modèle de base étant établi, n'importe quelle extension du produit pourrait être au préalable, aisément traitée et testée par simulation avant de passer au stade de la réalisation pratique.

CHAPITRE 5

SYSTEME LOGICIEL

Toute ébauche d'un produit, dans quelque domaine que ce soit, s'évalue au degré de son pragmatisme. Aussi, conscients sommes-nous de cette vérité, l'axe d'investigation de nos efforts, après une étude du modèle mathématique et une élaboration du produit simulé, se concentra sur la conception et la réalisation pratique du produit. Les deux phases précédentes sont nécessaires à l'accomplissement sans entraves, de cette étape, mais celle-ci reste le principal et ultime objectif. Elle représente l'aboutissement d'un effort et la concrétisation d'un concept.

La réalisation du produit pratique traverse deux étapes distinctes qui s'enchaînent. La première établit la charpente du produit: c'est la réalisation matérielle de celui-ci. Elle consiste en une carte électronique enfermée dans un boîtier à jonctions avec l'extérieur.

Quant à la seconde, elle constituera le cerveau de la charpente: c'est la réalisation logicielle. Elle permet le fonctionnement de l'appareil physique de manière programmée.

Ce logiciel doit assurer la gestion des communications de deux équipements informatiques, sur deux sites, éloignés, reliés à l'aide d'une ligne de transmission à longue distance unique.

1.- TEMPS PARTAGE [5]

La fonction d'un système en temps partagé est d'offrir à chaque utilisateur l'équivalent d'une machine individuelle tout en le faisant bénéficier de services communs: partage d'informations, communication entre usagers. En outre, grâce à la répartition des coûts entre un grand nombre d'usagers, on espère pouvoir offrir à chacun d'eux des services communs qui lui seraient inaccessibles individuellement.

Les problèmes posés par la conception d'un système en temps

partagé combinent donc, ceux d'un ordinateur individuel et ceux d'un système à transactions. On peut les classer comme suit:

— définition de la machine virtuelle offerte à chaque usager,
— partage et allocation des ressources physiques communes,
— gestion de l'information partagée et des communications.

Dans notre système, la machine virtuelle serait la voie basse vitesse, offerte à l'usager pour véhiculer l'information, à l'aller et au retour. La ressource physique commune serait la voie haute vitesse que se partagent les voies basse vitesse pour transmettre leurs informations. Et la gestion de l'information partagée et des communications est assurée par le logiciel régissant le système.

Notre système est donc un système en temps partagé pour l'usager d'un organe de communication unique qui est la voie composite.

Les qualités requises d'un système en temps partagé combinent également celles d'une machine individuelle et d'un système à transactions: disponibilité, fiabilité, sécurité, bonne exploitation des performances du matériel, qualité de l'interface et des services offerts aux usagers et facilité d'extension et d'adaptation.

2.- PARALLELISME DES PROCESSUS

Nous avons montré dans le paragraphe 2.1 du chapitre 4 l'opportunité de traiter les processus liés aux voies basse vitesse selon une boucle de test (par polling) et les processus liés à la voie haute vitesse par interruption. De plus, nous avons émis l'impossibilité de traiter les processus liés à la gestion de la file d'attente par interruption et nous avons suggéré de traiter le terminal de supervision par boucle de test.

De cette organisation découlait un parallélisme entre les processus:

— émission sur la voie haute vitesse (1 processus),
et — réception sur la voie haute vitesse (1 processus),
et — gestion du terminal superviseur (1 processus),
ou — réception sur une voie basse vitesse (1 processus),
ou — constitution de la trame d'une voie basse vitesse, (1 processus)
ou — émission sur une voie basse vitesse (1 processus).

Quant aux processus suivants, ils étaient traités de manière séquentielle:
— gestion du terminal superviseur (1 processus),
— réception sur une voie basse vitesse (8 processus),
— constitution de la trame d'une voie basse vitesse (8 processus),
— émission sur une voie basse vitesse (8 processus).

Seulement cette appellation de «processus» relevait plutôt du domaine de la simulation que de celui de l'informatique, où le terme processus garde la signification qu'on lui a connue en simulation (combinaison d'activités et d'événements) augmentée de nouvelles restrictions. Ces restrictions concernent notamment l'exclusivité du contexte de travail d'un processus (c'est le contexte du processus) qui fait que le passage d'un processus à un autre implique nécessairement le changement du contexte de ce dernier. Or les processus «de simulation» séquentiels que nous avons déterminés ne nécessitent aucun changement de contexte; ils constitueront par conséquent, une succession de procédures d'un même processus «informatique» que nous appellerons le processus boucle de test.

Cette nouvelle organisation nous fournit le parallélisme suivant:
— émission sur la voie haute vitesse (1 processus),
et — réception sur la voie haute vitesse (1 processus),
et — boucle de test (1 processus).

Le processus boucle de test, étant constitué des procédures:

— gestion du terminal superviseur (1 procédure),
— réception sur une voie basse vitesse (8 procédures),
— constitution de la trame d'une voie basse vitesse (8 procédures)
— émission sur une voie basse vitesse (8 procédures).

Afin d'activer le service d'un périphérique en cas de remplissage de sa FIFO de réception, lors d'une opération d'émission, nous pouvons envisager de le servir en priorité, par interruption. Ceci sera réalisé à l'aide des processus réception sur une voie basse vitesse avec récupération en bloc (8 processus), qui se chargeront de la récupération des trois entités émises.

Ces processus concurrenceront les processus d'émission, et de réception sur la voie haute vitesse.

Nous noterons qu'ils ne sont pas indispensables au système, puisqu'en cas de remplissage de la FIFO de réception, un signal est généré pour stopper l'émission sur la voie concernée.

D'autre part, il apparaît que le processus émission sur la voie haute vitesse peut s'auto-interrompre une ou plusieurs fois. Il peut donc, à lui tout seul, présenter deux ou plusieurs processus parallèles.

Il en est de même pour le processus réception sur la voie haute vitesse.

- Récapitulatif -

Les processus parallèles sont:

— émission sur la voie haute vitesse (2 processus au moins),
— réception sur la voie haute vitesse (2 processus au moins),
— réception sur la voie basse vitesse avec récupération en bloc (8 processus),
— boucle de test (1 processus).

Le décompte de tous ces processus donne au moins treize (13) processus parallèles.

3.- INTERRUPTIONS [5]

La mise en oeuvre pratique des processus simultanés est réalisée à l'aide des interruptions.

Les causes d'interruption peuvent être diverses. Le mécanisme d'interruption doit permettre de distinguer ces causes.
— Un indicateur distinct, appelé niveau d'interruption, est attaché à chaque cause d'interruption. A chaque niveau, correspond donc, un programme d'interruption distinct.
— Un indicateur est utilisé pour toutes les interruptions. Un programme de traitement unique lui est associé. Une information supplémentaire, contenue dans le mot d'état ou dans un registre, permet de distinguer entre les causes possibles.

Le processeur que nous avons utilisé offre trois niveaux

d'interruptions matérielles: NMI, IRQ et FIRQ. Nous en avons utilisé les deux premières. L'ordre de priorité accordé à ces interruptions de manière décroissante est le suivant: NMI, FIRQ et IRQ.

La NMI est donc l'interruption la plus prioritaire. De plus, elle n'est pas masquable; pour empêcher son arrivée, on doit la désarmer. Pour ses propriétés, nous avons octroyé cette interruption aux processus d'émission et de réception sur la voie haute vitesse, processus nettement plus rapides que les autres, et qui, par conséquent requierent un temps de réponse très bref. Seulement, nous remarquerons, que ce niveau d'interruption est d'ores et déjà attribué à deux causes distinctes: l'émission et la réception. La distinction entre les deux causes (ou processus) se fera à l'aide d'une simple scrutation de registre se trouvant dans l'interface qui déclenche l'interruption dès qu'une émission ou réception de caractère est prête à se faire. Nous verrons par la suite, que ce même niveau d'interruption est attribué à d'autres causes d'interruption encore, que l'émission et la réception, mais ces causes sont toujours liées à la voie haute vitesse.

Quant aux huit (8) processus réception sur une voie basse vitesse avec récupération en bloc ils seront traités par l'interruption masquable IRQ. Là aussi, le niveau d'interruption sera attribué à 8 causes (ou processus) différentes. De même que précédemment, la distinction entre ces causes se fera après consultation de quatre registres situés dans l'interface relié aux voies basse vitesse. Chaque registre (correspondant à deux voies), nous informera sur deux causes.

Nous avons opté pour l'octroi de la NMI à la voie composite et l'IRQ aux voies basse vitesse, car rajouté à la rapidité du débit de l'information sur la voie haute vitesse comparativement à celui des voies basse vitesse, l'IRQ ne sera sollicitée qu'occasionnellement: lorsque le processeur n'aura pas le temps de servir en un laps de temps très réduit, la voie haute vitesse et les 8 voies basse vitesse, et que par conséquent, la FIFO de l'une des 8 voies se remplit.

Ce système d'interruption nous permet donc de traiter les douze (12) processus suivants, de manière parallèle:
— émission sur la voie haute vitesse,
— réception sur la voie haute vitesse,
— émission sur une voie basse vitesse avec récupération en bloc.

Le treizième processus qui leur est parallèle (boucle de test) sera traité de manière bouclée.

4.- STRUCTURE DE DONNEES

Les données du système sont exclusivement représentées par l'information échangée. Le reste constitue la file d'attente des trames à véhiculer sur la voie haute vitesse et les variables de travail.

4.1.- Mémoires tampons

Ces mémoires recueillent l'information échangée à l'émission et à la réception (voir figure 5.1). A partir de cette simple définition, on peut d'ores et déjà distinguer entre deux catégories de mémoire tampon (ou buffer): une catégorie recueillant l'information émise des voies basse vitesse pour être transmise sur la voie haute vitesse: c'est la mémoire tampon de réception; et une autre catégorie récupérant l'information déversée par la voie haute vitesse et destinée à être dirigée sur les voies basse vitesse: c'est la mémoire tampon d'émission.

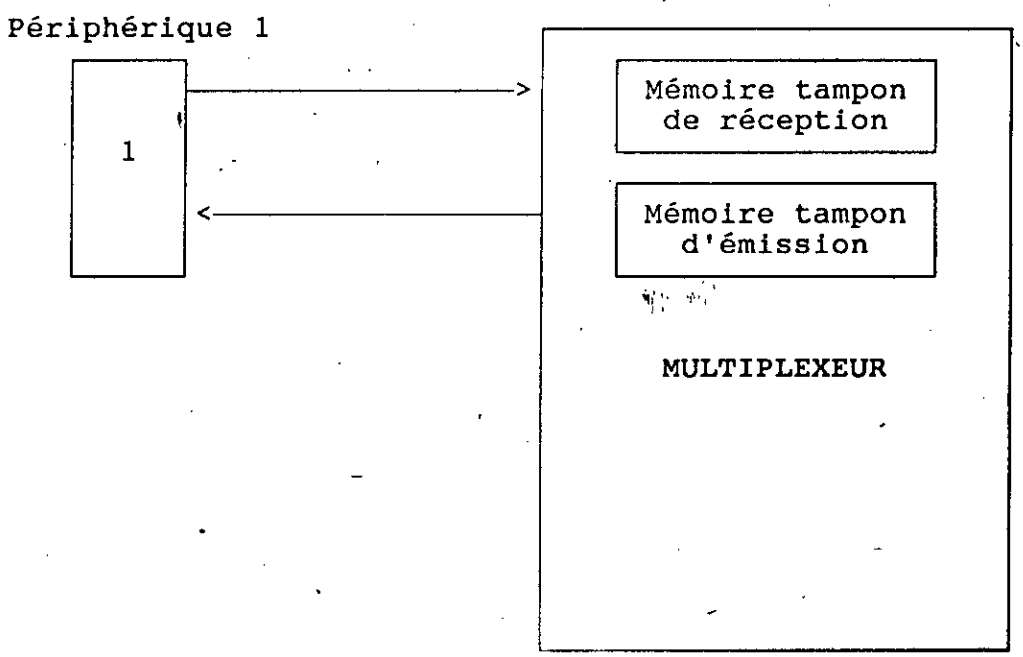


Figure 5.1. Mémoires tampons

De plus, nous disposons de huit (8) voies distinctes; il serait donc indiqué de départager les deux catégories de mémoire, chacune en huit (8) zones différentes. Le décompte de l'ensemble de ces zones nous mène à seize (16) mémoires tampons, chacune indiquée par le numéro de la voie (i = 1,8) et le sens de la transmission (j = 1,2).

Pour une exploitation efficace, maximum et de mise en oeuvre facile de l'ensemble de ces mémoires, nous avons pensé à la structure de file circulaire. Ainsi tout l'espace vide peut être utilisé à tout moment. Quant à la gestion du buffer, elle se limitera à la gestion de deux pointeurs de tête et de queue de la mémoire.

De part sa définition de file, cette mémoire sera, bien entendu, servie selon le principe «premier arrivé, premier servi».

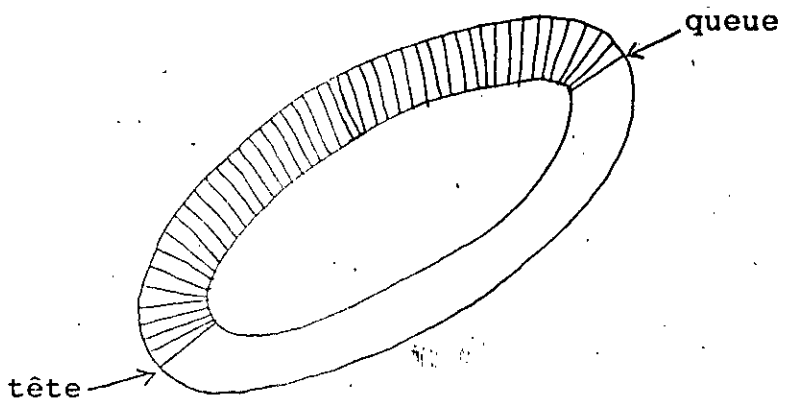


Figure 5.2. Structure circulaire de la mémoire tampon

Lors de la mise en oeuvre pratique de ces mémoires, sachant que l'espace mémoire physique est consécutif (et non circulaire), nous tiendrons compte des deux cas de figure de mémoire suivants:

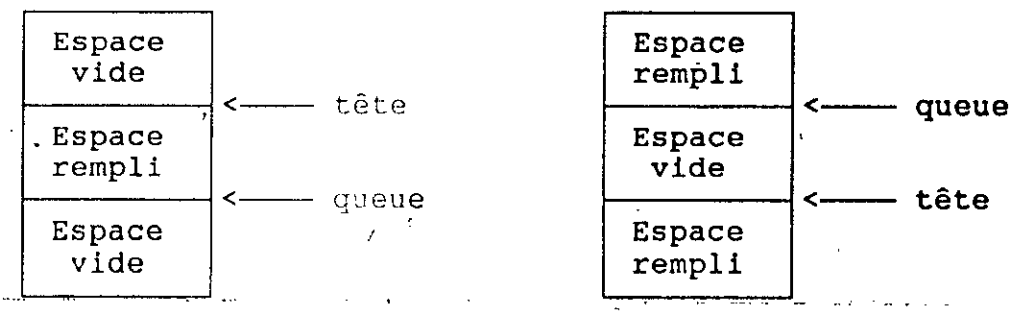


Figure 5.3. Structure physique de la mémoire tampon

Par ailleurs, le traitement de ces mémoires étant le même à l'émission ou à la réception pour les huit (8) voies, nous avons pensé utiliser l'adressage indexé afin d'optimiser la taille du programme (même procédure pour les huit (8) voies) et activer l'accès à la mémoire en évitant le test sur la voie concernée. Les adresses des mémoires tampons d'émission ou de réception sont donc placées dans un tableau dont l'indice correspond au numéro de la voie à laquelle est affectée la mémoire.

Tableau 5.1. Table des adresses des mémoires tampons des 8 voies

Indice	Adresses
1	adresse de la mémoire tampon de réception de la voie 1
2	adresse de la mémoire tampon de réception de la voie 2
.	.
.	.
.	.
8	adresse de la mémoire tampon de réception de la voie 8

Nous utilisons le même procédé pour adresser les pointeurs de ces mémoires et leurs compteurs de caractères.

4.2.- File d'attente

Cette file met en attente les trames prêtes à être émises sur la voie haute vitesse. Ces trames ne sont rien d'autre que des séquences de l'information échangée; or cette information, on l'a vu, est emmagasinée dans les mémoires tampons de réception. Les données de la file d'attente constitueront donc un ensemble d'informations sur les trames.

L'une des informations essentielles est le numéro de la voie d'où provient la trame. Une autre information importante pour la constitution de la file d'attente, est l'adresse de la trame en mémoire tampon. Enfin, la dernière information porte sur le nombre de caractères de la trame, déjà émis sur la voie haute vitesse, vu que l'émission sur cette voie ne se fait pas en bloc mais plutôt, caractère sur caractère moyennant le processus émission sur la voie haute vitesse.

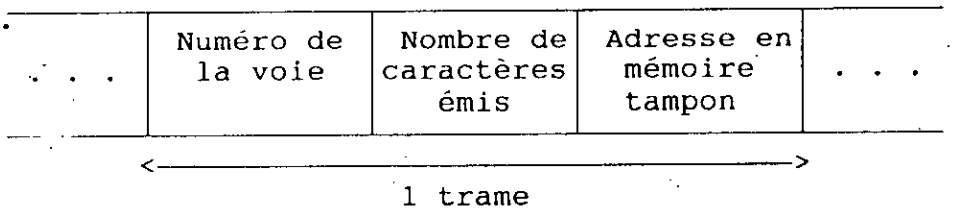


Figure 5.4. Structure de la file d'attente

Cette mémoire est servie selon le principe «premier arrivé, premier servi». Elle présente une structure de file circulaire régie par deux pointeurs de tête et de queue de mémoire. La mise en oeuvre pratique (structure physique) est similaire à celle de la mémoire tampon.

4.3.- Régulation du flot des données

Grâce à la présence des mémoires tampons, le débit maximum cumulé des voies basse vitesse peut être supérieur au débit de la voie composite, d'où une efficacité supérieure à 1 (ou un taux de concentration de plus de 100%).

$$@ = D^{-1} \sum_{1}^{N} di$$

@ = efficacité,
di = débit de la voie basse vitesse i,
D = débit de la voie haute vitesse.

Ici, l'on s'impose:

$$\sum_{1}^{8} di = 19.2 \text{ KB}$$

d'où: @ = 19.2/9.6 = 200 %

Mais ceci présente un risque de saturation des mémoires tampons, et donc une perte d'information si tous les terminaux sont actifs en même temps. Pour pallier à ce problème, on procède donc à un contrôle de flux; dès que la mémoire tampon est remplie à 80 % on arrête le débit des terminaux à l'aide d'un XON/XOFF ou de signaux CTS/RTS.

5.- DESCRIPTION DES PROCESSUS

Les processus du système ont été présentés et définis dans le paragraphe 2.1 du chapitre précédent et réajustés dans le paragraphe 2 du présent chapitre. Nous tâcherons, à présent, de les décrire, processus et procédures confondus, dans l'ordre dans lequel ils se déroulent, bien qu'un certain nombre d'entre eux s'exécute de manière parallèle; mais en fait, ceci ne présente aucune contrainte, car au lancement du système, ils arrivent tous de manière séquentielle, vu que certains d'entre eux sont activés par d'autres. Exemple: l'émission sur la voie haute vitesse active la réception sur cette voie.

5.1.- Gestion du superviseur

Le terminal superviseur réalise l'établissement d'une liaison entre deux points de deux sites éloignés.De plus, il assure la configuration de l'appareil de travail local (terminal,ordinateur...).

En effet, après quelques écrans de mise en service du multiplexeur, le terminal superviseur offre aux usagers la possibilité de connexion logique à distance par simple introduction du numéro de la voie locale à mettre en service. Le message suivant est alors affiché:

Entrez le numéro de votre voie (1 à 8):

Après réception du numéro de la voie, le superviseur s'enquiert d'une éventuelle reconfiguration de l'appareil connecté à la voie mise en service (une configuration automatique est établie par défaut). Le message suivant apparaît sur écran:

Désirez-vous une reconfiguration ? (O ou N):

Si l'usager répond oui (O), le menu de la reconfiguration de son appareil apparaît sur écran, paramètre par paramètre. Le premier paramètre concerne la vitesse d'émission avec une vitesse affichée par défaut. L'utilisateur peut, moyennant la touche «flèche droite», changer à son gré, cette vitesse. Donc, à partir de la vitesse affichée, à chaque fois qu'il tapera «flèche droite», une nouvelle vitesse apparaîtra. L'ensemble des vitesses offertes est: 1200, 2400, 4800, 9600, 75, 150, 300 et 600 ; la vitesse 1200

étant celle affichée par défaut.

Lorsque l'usager parvient à la vitesse qui lui convient, il doit la confirmer, au moyen de la touche «Return». A ce moment, le paramètre suivant est affiché. Et le processus reprend avec chaque nouveau paramètre. Au terme du dernier paramètre, nous retrouvons l'écran suivant:

```
Vitesse d'émission                  :     1200
Vitesse de réception                :     1200
Nombre de bits par caractère        :     8
Parité                              :     non
Nombre de bits STOP                 :     2
```

Les valeurs des paramètres affichées ici, sont celles prises par défaut.

Dans le tableau 5.2 nous retrouvons la liste des valeurs de chacun de ces paramètres.

TAbleau 5.2. Paramètres de configuration d'un terminal

Vitesse d'émission	Vitesse de réception	Nombre de bits par caractère	Parité	Nombre de bits STOP
1200 2400 4800 9600 75 150 300 600	1200 2400 4800 9600 75 150 300 600	8 5 6 7	NON PAIRE IMPAIRE FORCEE A 0 FORCEE A 1	2 1

Lorsque la valeur du dernier paramètre est introduite, le superviseur est prêt à mettre en service une nouvelle voie.

Quant à la configuration proprement dite du périphérique qui vient d'être traité, elle est réalisée par une procédure qui récupère les paramètres introduits, les analyse, les décode, les convertit à la forme adéquate à la structure des registres appropriés de l'OCTAL-UART, puis les stocke dans ces registres.

La configuration du superviseur, lui-même, est assurée par des interrupteurs situés sur la carte multiplexeur. Ces interrupteurs sont lus à travers un port d'entrée/sortie. Les valeurs lues sont traitées, puis stockées dans les registres appropriés de la voie B du DUSART, dans la procédure de configuration du superviseur. Cette procédure est exécutée à la mise sous tension. Elle ne fait donc pas partie du processus **gestion du superviseur**, qui peut lui, revenir à chaque fois que l'on veut établir une liaison.

Aussi, grâce à la propriété de parallélisme de ce processus, le superviseur offre la possibilité de connexion à distance, à tout moment, à partir de la mise en service du multiplexeur, sans pour autant perturber les communications déjà établies.

5.2.- Réception sur une voie basse vitesse

Ce processus se charge de l'acquisition d'un caractère arrivant sur la voie i. Cette acquisition est réalisée après test du bit de réception d'un caractère, situé dans le mot d'état de l'UART relatif à la voie. Le caractère est placé dans la mémoire tampon, le pointeur de tête du buffer remis à jour et un compteur du nombre de caractères de la mémoire tampon est incrémenté. Si le buffer est rempli à plus de 80 %, le débit de l'appareil connecté sur la voie est stoppé momentanément. Dès que l'appareil s'arrête d'émettre, un processus de comptage est déclenché, pour signaler une éventuelle fin d'émission. Les signaux de mise en service de la ligne et de l'appareil (CTS, RTS, DCD,...) sont contrôlés et gérés par la procédure.

5.3.- Constitution de la trame d'une voie basse vitesse

A chaque fois que l'on dispose de suffisamment d'information dans la mémoire tampon pour constituer une trame, une nouvelle entrée dans la file d'attente de ces trames est réalisée. La quantité de cette information est fixe (trames de longueur fixe), sauf lorsqu'il s'agit de la dernière trame émise. Elle se chiffre en caractères.

Une autre tâche importante de ce processus, est celle de lancer le processus **émission sur la voie haute vitesse**, quand la première trame est constituée, puis toutes les fois que l'émission

sur la voie haute vitesse est interrompue pour manque d'information émanant de la voie basse vitesse.

5.4.- Emission sur la voie haute vitesse .

Cette procédure se charge de l'émission, sur la voie haute vitesse, d'un caractère d'une trame déjà constituée. Ce caractère peut être le numéro de la station secondaire (voir paragraphe 6), le numéro de la voie du correspondant ou de l'information propre. Dans ce dernier cas, le pointeur de queue et le compteur de caractères de la mémoire tampon de réception sont mis à jour. Quand les caractères d'une trame donnée sont épuisés, celle-ci est défilée de la file d'attente et une nouvelle trame (si présente dans la file) est désignée pour l'émission.

L'organisation et la gestion proprement dites de cette émission, notamment au niveau de la trame, sont régies par le protocole SDLC, développé dans le paragraphe 6.2. L'exploitation de ce protocole est réalisée par la présente procédure.

5.5.- Réception sur la voie haute vitesse

Ce processus réalise la réception d'un caractère provenant de la voie haute vitesse. Ce caractère peut être le numéro de la station secondaire, le numéro de la voie destinataire ou de l'information propre. Moyennant le numéro du destinataire, cette information est dirigée vers la mémoire tampon correspondant au destinataire. Le pointeur de tête de la mémoire est remis à jour.

L'organisation de cette réception, notamment au niveau de la trame, est régie par le protocole SDLC développé dans le paragraphe 6.3. L'exploitation de ce protocole est également réalisée par ce processus.

5.6.- Emission sur une voie basse vitesse

Si la mémoire tampon d'émission n'est pas vide, un caractère est transféré sur la voie basse vitesse. Le pointeur de queue de la mémoire est mis à jour.

6.- PROTOCOLE DE COMMUNICATION [10]

De part et d'autre de la voie haute vitesse, à travers l'interface de communication DUSART, sont connectés deux multiplexeurs. Ces deux multiplexeurs communiquent à l'aide du protocole SDLC qui leur offre la facilité et la souplesse de la communication grâce aux règles préétablies, régissant l'émission et la réception sur cette voie.

Le message SDLC ou trame (voir figure 5.5) débute et finit par un drapeau. Ce drapeau est reconnu par l'interface de communication comme délimiteur de trame. Notons que le drapeau peut être reçu par l'interface mais jamais émis, grâce à l'insertion automatique du zéro, après une séquence de cinq (5) «1», par l'émetteur et sa suppression par le récepteur.

Les huit (8) bits du champ d'adresse du message SDLC sont destinés à recevoir l'adresse de la station secondaire. Le récepteur est doté d'un mode de recherche lui permettant de relever, sur la trame, cette adresse, la comparer avec l'adresse de la même station, contenue dans un registre et rejeter la trame si le résultat de la comparaison donne deux adresses différentes.

Le champ de commande de la trame est transparent pour l'interface. Il est simplement transféré à l'unité centrale.

Les lignes de transmission ne sont pas parfaites; elles introduisent des erreurs sur certains symboles binaires. Le taux d'erreur étant inacceptable pour beaucoup d'applications, on pallie ces erreurs en ajoutant à l'information utile à transmettre une quantité d'information supplémentaire dite "redondante". La redondance utilisée dans notre cas sera représentée par deux caractères CRC (code de redondance cyclique). Elle servira à détecter l'erreur.

Drapeau d'ouverture 01111110	Adresse 8 Bits	Champ des données	CRC #1	CRC #2	Drapeau de fermeture 01111110

Figure 5.5. Format de la trame

La mise en oeuvre de l'interface, moyennant le protocole SDLC, nécessite l'initialisation des registres suivants: le Registre de Contrôle de Modes, le Registre de Contrôle d'Interruptions, le Registre de Contrôle de Réception, le Registre de Contrôle d'Emission, le Registre de Mot de Synchronisation 1 et le Registre de Mot de Synchronisation 2. Le Registre de Contrôle de Modes doit être programmé avant les autres registres. Quant aux registres suivants, ils assurent le transfert des données ou la communication d'états entre l'interface et le processeur: le Registre de Commandes, le Registre d'Etat 0, le Registre d'Etat 1, le Registre de Données et le Registre de Vecteur d'Interruption.

6.1.- Initialisation

L'interface est initialisé au mode SDLC après sélection des paramètres suivants, dans le Registre de Contrôle de Modes: X1 Mode Horloge, Mode SDLC et Mode de Synchronisation Actif. La parité n'est pas utilisée dans le Mode SDLC car l'émetteur ne rajoutera pas le bit de parité au drapeau ni aux deux caractères CRC, et ceci causera une erreur de parité au niveau du récepteur.

Comme le CRC doit être calculé pour la donnée émise, le paramètre SDL-CRC sera sélectionné dans le Registre de Contrôle d'Interruptions.

Le Registre de Mot de Synchronisation 1 contiendra l'adresse de la station secondaire et le Registre de Mot de Synchronisation 2 renfermera le drapeau qui sera programmé à 01111110.

Les paramètres CRC de Réception Actif et Mode de Recherche d'Adresse, du Registre de Contrôle de Réception, doivent aussi être utilisés. Le premier valide le calcul du CRC au niveau du récepteur et le second active la recherche de l'adresse de la station secondaire sur la trame.

Enfin, le paramètre CRC d'Emission Actif est initialisé dans le Registre de Contrôle d'Emission pour valider le calcul du CRC à l'émission.

Toutes ces initialisations (relatives au protocole SDLC), sont réalisées dans la procédure d'initialisation de l'interface.

6.2.- Emission SDLC

L'émission SDLC n'est rien d'autre que l'émission sur la voie haute vitesse, processus dont les fonctions ont été bien définies dans le paragraphe 2. Il reste donc à préciser l'exploitation du protocole dans ce processus d'émission.

6.2.1.- Début d'émission

Une fois que l'émission a été validée par le paramètre Emission Active dans le Registre de Contrôle d'Emission, elle est déclenchée après chargement du premier caractère dans son Registre d'Emission (Registre de Donnée).

Pour un calcul correct du CRC tous les bits du générateur de CRC doivent être initialisés à 1, avant tout chargement de caractère dans le Registre de Réception, pour chaque nouvelle trame. Ceci, est accompli par simple sélection de la commande Initialisation du Générateur de CRC à l'Emission dans le Registre de Commandes. Tous les caractères compris entre les drapeaux d'ouverture et de fermeture de la trame, sont inclus dans le calcul du CRC. La sortie du générateur de CRC est inversée avant sa transmission.

Le premier caractère émis différent du drapeau est l'adresse de la station secondaire. L'interface ne se charge pas de son émission qui est laissée à la charge du programmeur. Les seuls caractères transmis automatiquement sont les drapeaux et les CRC.

6.2.2.- Emission des données

Une interruption d'émission est générée à chaque fois que le Registre d'Emission est vide. Ceci est rendu possible grâce à la programmation du paramètre Interruption d'Emission Active, dans le Registre de Contrôle d'Interruptions. Cette interruption doit être satisfaite soit par écriture d'un nouveau caractère dans le Registre d'Emission soit par la commande Réactiver l'Interruption d'Emission. Si l'interruption est satisfaite par cette commande et aucun autre caractère n'est écrit dans le Registre d'Emission, aucune autre interruption ne surviendra et la condition de fin de trame apparaît, dès que la donnée contenue dans le Registre à

Décalage est transmise. Quand un nouveau caractère est chargé dans le Registre d'Emission, ce dernier peut à nouveau redevenir vide et par conséquent interrompre le processeur.

En effet, ce problème peut surgir dans notre système, quand on est momentanément à court d'information à émettre, sans que pour cela, l'émission soit achevée. Pour pallier à cela, nous saisissons dans la procédure **constitution de la trame d'une voie basse vitesse** du processus **boucle de test**, l'événement de la constitution d'une trame pour tester un éventuel arrêt momentané de l'émission et la relancer dans le cas affirmatif. Nous utilisons le même procédé pour démarrer l'émission car tout au début, le registre d'émission est déjà vide, et ne peut donc pas se vider générant ainsi une interruption. En d'autres termes, une interruption d'émission ne survient que lorsque le registre d'émission déjà chargé, vient de se vider.

Une autre méthode, pour détecter une demande de service de l'émetteur, serait de tester de manière bouclée le bit Registre d'Emission Vide dans le Registre d'Etat.

6.2.3.- Détection d'une fin de trame

L'interface termine automatiquement une trame quand le Registre d'Emission est vide et le Registre à Décalage ne contient plus de bits à envoyer: c'est la condition de fin de trame. Cette technique permet une grande vitesse d'émission en déchargeant le processeur de la tâche de répondre rapidement à chaque situation de fin de trame.

La terminaison de la trame (2 CRC et un drapeau ou un drapeau seulement) dépend de l'état du bit Fin de Trame Emise dans le Registre d'Etat 0. A la suite d'une initialisation du système, ce bit est mis à 1 pour prévenir toute insertion du caractère CRC tant qu'aucune donnée n'est à émettre. Par conséquent, seuls les drapeaux sont émis bien que l'on se trouve dans une condition de fin de trame. Si par contre ce bit est à zéro (0) lors d'une situation de fin de trame, les 16 bits du caractère CRC sont envoyés suivis d'un ou de plusieurs drapeaux. Ce bit est mis à zéro à l'aide de la commande Fin de Trame avec CRC dans le Registre de Commandes. Aucune restriction n'est à signaler quant au moment où le bit, Fin de Trame Emise dans le Registre d'Etat 0, doit être mis

à zéro, au cours de la transmission de la trame. Il est d'habitude mis à 1 après chargement du premier caractère (adresse de la station secondaire) dans le Registre d'Emission.

L'émission du premier bit du caractère CRC, remet le bit Fin *de Trame* du Registre d'Etat 0 à 1 et génère une Interruption Externe si cette source d'interruption a déjà été validée par le paramètre Interruption Externe Valide dans le Registre de Contrôle d'interruption. Cette interruption doit être satisfaite par la commande Réactiver Interruption Externe dans le Registre de commandes. Aussi tant que le caractère CRC (16 bits) est en cours d'émission, le bit Registre d'Emission Vide dans le Registre d'Etat 0 signale que le Registre à Décalage est plein de la donné CRC (bit mis à 0). Dès que le CRC est entièrement émis, ce bit est remis à 1, et une Interruption d'Emission est générée, signalant la possibilité d'écriture d'une nouvelle donnée dans le Registre d'Emission.

L'Interruption Externe a été exploitée par notre système pour entamer l'émission d'une nouvelle trame: si l'on se basait uniquement, sur la longueur de la trame et la disponibilité du Registre d'Emission, on risquerait de lancer une nouvelle trame alors que les CRC de la trame précédente n'auraient pas encore été envoyés.

Cette interruption se situe au niveau de l'interruption NMI, pour laquelle elle constitue avec les deux autres causes (émission et réception sur la voie haute vitesse), déjà vues dans le paragraphe 3, une nouvelle cause.

6.3.- Réception SDLC

La réception SDLC n'est rien d'autre que la réception sur la voie haute vitesse, processus dont les fonctions ont été définies dans le paragraphe 5.2. Il reste donc à préciser l'exploitation du protocole dans ce processus de réception.

6.3.1.- Réception des données

Une Interruption de réception est générée à chaque fois qu'un caractère est reçu dans la FIFO de réception. Cette interruption est validée à l'aide du paramètre Interruption à Chaque Caractère Reçu (parité non considérée) dans le Registre de Contrôle d'Inter-

ruptions. Elle est satisfaite après lecture du Registre de Réception.

La donnée arrive dans la FIFO de réception accompagnée de son état qui peut être l'un des trois suivants: Débordement en Réception, Erreur de Parité ou Fin de Trame Reçue. On dit alors que l'on est dans une situation de Réception Spéciale. Cette situation est traduite par le positionnement des trois bits de poids faible du Registre de Vecteur d'Interruption à la valeur 7.

6.3.2.- Détection d'une fin de trame

Une trame est terminée par le chargement du deuxième caractère CRC dans la FIFO de réception. Ce chargement positionne le bit Fin de Trame Reçue dans le Registre d'Etat 1 à 1 et génère une Interruption de Réception Spéciale. Le caractère CRC doit être lu puis écarté. Cependant, ne disposant pas de moyen permettant de repérer le premier CRC, ce dernier est stocké dans la mémoire tampon d'émission comme un quelconque caractère, puis retiré à la détection du deuxième CRC. Une précaution reste cependant à prendre: ne jamais émettre ce caractère sur la voie basse vitesse, en attendant de le retirer de la mémoire tampon.

7.- SYNCHRONISATION DES PROCESSUS [5, 6, 7]

La conception et la programmation d'applications se trouvent actuellement influencées par l'utilisation de la programmation modulaire et parallèle. Ceci conduit à exprimer une application comme un ensemble de processus relativement indépendants les uns des autres et non plus comme un programme dont l'exécution est séquentielle. Ces processus ont nécessairement des relations de coopération entre eux.

De plus, la mise en oeuvre de plusieurs processus parallèles à l'aide d'un nombre limité de ressources (mémoires, unités centrales, etc.), pose des problèmes de compétition.

Ces problèmes de coopération et de compétition sont résolus à l'aide de différents moyens de synchronisation qui consistent à cadencer l'évolution des processus, en introduisant une relation d'ordre dans leur exécution.

Un processus est généralement constitué d'une suite d'événements. L'événement peut être local au processus ou, au contraire, significatif pour l'ensemble du système. Il concerne alors, le problème de synchronisation posé.

7.1.- Détermination des ressources critiques

Une ressource critique est une ressource disputée par plusieurs processus parallèles, Ces processus sont alors en exclusion mutuelle pour l'usage de la ressource.

7.1.1.- Partage du processeur

Dans notre cas, l'une des ressources les plus disputées est le processeur, vu son unicité devant la multiplicité des processus parallèles.

7.1.2.- Partage de la mémoire

La mémoire constitue souvent une ressource critique de taille. En effet, pour notre système, les mémoires de stockage temporaire de l'information, appelées mémoires tampons ou buffers de stockage, peuvent être sollicitées simultanément par deux ou plusieurs processus parallèles. Les pointeurs de ces mémoires peuvent être eux aussi, des zones mémoires communément sollicitées par des processus parallèles.

De même, les compteurs de caractères de ces buffers et les registres de travail des voies basse vitesse sont souvent à l'origine de conflits entre processus parallèles.

Et enfin, de manière générale, une simple variable de travail peut constituer une source d'exclusion mutuelle entre processus.

7.1.3.- Partage de programme

Deux programmes étant identiques, on peut envisager de n'en conserver qu'une copie unique en mémoire. Les conditions nécessaires à ce partage de programme sont:

— le texte du programme ne doit pas être modifié par son exécution,
— lorsque le programme s'exécute pour le processus Pi, il doit accéder au segment de données Di; la désignation indirecte, est tout indiquée pour cet adressage sélectif des données. Le programme désigne ses données par l'intermédiaire d'un registre de base qui contient selon le cas, l'adresse d'implantation du segment Di. Seul ce registre doit être chargé à chaque commutation, au même titre que les registres internes ou le compteur ordinal.

Cette ressource est également présente dans notre système. Elle est représentée par la portion de programme qui traite la récupération d'un bloc de trois caractères émis à travers une voie basse vitesse; ce programme est sollicité par les huit (8) processus **réception sur une voie basse vitesse avec récupération en bloc**. Ici, la désignation indirecte est double. La première porte sur le numéro de la voie et la deuxième sur l'adresse de la donnée (qui est fonction du numéro de la voie).

Nous pouvons aussi citer l'exemple de programme appelé deux ou plusieurs fois simultanément par le même processus. C'est le cas particulier d'un processus interrompu par lui-même; nous donnons comme exemple les processus **émission sur la voie haute vitesse** et **réception sur la voie haute vitesse**.

7.2.- Construction des sections critiques

Une section critique est un évènement du processus sollicitant une ressource critique.

Cet événement peut parfois s'étendre à un ensemble d'événements successifs constituant une grande partie du processus. C'est le cas où plusieurs ressources partageables sont traitées dans le même processus. Dans notre système, nous pouvons citer l'exemple de:
— la procédure d'émission d'un caractère, qui appartient au processus **émission sur la voie haute vitesse**,
— la procédure de réception d'un caractère, qui appartient au processus **réception sur la voie haute vitesse**.

Dans le paragraphe précédent, nous avons déterminé les mémoires pouvant constituer une ressource critique pour notre

système. Nous tâcherons de construire dans ce paragraphe des exemples de sections critiques relatives aux ressources précitées.

7.2.1.- Mémoires tampons

Ces mémoires stockant l'information transmise sur les voies basse vitesse à l'aller et au retour sont classées parmi les problèmes du producteur-consommateur.

En effet, soit la mémoire tampon d'émission Mi, contenant l'information à émettre sur la voie basse vitesse i. Le remplissage de cette mémoire est assuré par le processus: **réception sur la voie haute vitesse**. Quant à son vidage, il est pris en charge par la procédure: **émission sur une voie basse vitesse** du processus **boucle de test**. Ces deux processus sont parallèles; ils peuvent, par conséquent, solliciter simultanément la mémoire Mi. La section critique serait donc l'événement du processus sollicitant la ressource Mi en remplissage ou en vidage. Les précautions à prendre seraient de ne point remplir une mémoire déjà pleine, et de ne point vider une mémoire déjà vide.

Ici, la synchronisation semble être relativement simple à mettre en oeuvre.

7.2.2.- Pointeurs de mémoires tampons

La mise à jour d'un pointeur de mémoire tampon peut constituer une section critique pour les processus concernés. Il apparaît dans le paragraphe précédent que les mémoires tampons sont remplies et vidées par des processus concurrents. Ces processus, après remplissage ou vidage de la mémoire, procèdent à une mise à jour des pointeurs. Il s'agit bien sur, de faire attention à ne pas lire la valeur d'un pointeur qui n'a pas encore été mise à jour. C'est le problème classique du lecteur-rédacteur qui se pose.

Cette ressource peut également, être sollicitée par la procédure: **réception sur une voie basse vitesse** du processus **boucle de test**, et le processus **réception sur une voie basse vitesse avec récupération en bloc**. Le même problème que précédemment se pose, lors de la mise à jour du pointeur, avec la seule différence, qu'ici, il s'agit de la mise à jour d'un pointeur après deux

remplissages simultanés, et non plus un remplissage et un vidage.

7.2.3.- Compteurs

Le compteur des caractères de la mémoire tampon de réception relative à une voie basse vitesse i, est traité dans les processus concurrents suivants:
— la procédure **réception sur une voie basse vitesse** du processus **boucle de test**,
— le processus **réception sur une voie basse vitesse avec récupération en bloc**,
— le processus **émission sur la voie haute vitesse**.

Sa mise à jour pose le problème du lecteur-rédacteur, et devrait donc se faire non sans multiples précautions.

7.2.4.- Registres

L'utilisation des registres de travail d'une voie basse vitesse i, peut constituer une section critique. Leur mise à jour est traitée dans les processus parallèles: **boucle de test** (à l'aide de la procédure **réception sur une voie basse vitesse**) et **réception sur une voie basse vitesse avec récupération en bloc**. Donc, si l'on ne fait pas suffisamment attention pour ne pas considérer l'état d'un registre ne correspondant pas au processus voulu, un problème de cohérence se pose. C'est le problème du lecteur-rédacteur.

7.2.5.- Variables

Enfin, une simple variable de travail peut être une source d'exclusion mutuelle entre processus. Dans notre système, cette variable nous informe sur le type d'information à émettre sur la voie haute vitesse: numéro de la station secondaire, numéro de la voie du correspondant ou message d'information. Cette variable est utilisée dans les processus parallèles **émission sur la voie haute vitesse** et **boucle de test** (à l'aide de la procédure **constitution de la trame d'une voie basse vitesse**). C'est encore une fois, le problème du lecteur-rédacteur qui se pose pour la gestion de cette variable.

7.3.- Mécanisme de synchronisation

7.3.1.- Principe [6]

L'opération de synchronisation des processus est établie pour solutionner le problème posé par l'exclusion mutuelle de ces processus pour une ressource partageable.

Un moyen de synchronisation efficace, consiste à imposer au processus concurrent d'«attendre», pour rentrer en section critique, qu'une certaine condition soit satisfaite. Pour exprimer cette attente, nous introduisons pour le processus un nouvel état, pour lequel il sera dit **en attente**, ou **bloqué**, par opposition à l'état considéré implicitement jusqu'ici, où le processus est dit actif. Ainsi, la transition actif-attente sera appelée **blocage** et la transition inverse, **réveil**.

La notion d'attente sera donc utilisée pour spécifier la synchronisation entre processus. Cette spécification se fait en deux étapes:

1- Définir, pour chaque processus, ses points de synchronisation situés aux deux extrêmes limites de la section critique,
2- Associer à chaque point de synchronisation une condition de franchissement, exprimée au moyen de variables d'état du système.

Le problème de l'exclusion mutuelle se résoudrait donc par:

```
exclusion mutuelle (section critique) : début
                                        entrée;
                                        section critique;
                                        sortie
                                        fin
```

où: entrée et sortie sont des conditions de franchissement et section critique est une suite d'instructions utilisant la ressource critique.

7.3.2.- Mise en oeuvre [6, 7]

Une mise en oeuvre pratique de la solution précédente peut être traduite par l'utilisation de sémaphores.

Un sémaphore S est constitué d'une variable entière e(S) et

d'une file d'attente f(S). La variable e(S) peut prendre des valeurs entières positives, négatives ou nulles; nous l'appellerons simplement la valeur du sémaphore. La politique de gestion de la file d'attente est laissée à la guise du concepteur du système.

Un sémaphore S est crée par une déclaration qui doit spécifier la valeur initiale e0(S) de e(S). Cette valeur initiale est nécessairement un entier non négatif. A la création d'un sémaphore, sa file f(S) est toujours initialement vide.

On peut agir sur un sémaphore S par les deux seules primitives:

```
— P (s) : début
             e(s) = e(s) - 1 ;
             si e(s) < 0
                alors mettre le processus dans la file f(s)
             fsi;
          fin

— V (s) : début
             e(s) = e(s) + 1 ;
             si e(s) ≤ 0
               alors sortir un processus de la file f(s)
             fsi;
          fin
```

Les primitives sont indivisibles, ce qui assure que le test dans l'instruction conditionnelle et l'opération qui en découle se font toujours consécutivement.

Nous remarquerons que ces deux primitives ne sont rien d'autre que les conditions de franchissement d'une section critique autrement dit: «entrée» et «sortie» de la solution «exclusion mutuelle».

Un processus qui exécute P(S) peut se trouver bloqué si e(S) est négative ou nulle. Dans ce cas, seule une opération V(S) peut le libérer. Quand e(S) est négative, sa valeur absolue représente le nombre de processus en attente, quand elle est positive, elle indique le nombre maximum d'exécutions simultanées possibles.

La programmation d'une séquence en exclusion mutuelle est alors la suivante:

```
Exclusion mutuelle (section critique) : début
                                        P(S);
                                        SC;
                                        V(S)
                                        fin
```

8.- ALGORITHMES

8.1.- Réception d'un caractère sur une voie basse vitesse:

POLRBV

```
début
i = 1
tant que VOICON(i) ≠ CR
faire
   |   si BCTS(VOICON(i)) = 0          * CTS haut
   |   | alors appel RTSBAS
   |   |    |   si  C = 0  * RTS bas
   |   |    |   |   alors  aller à sortie
   |   |    |   |   sinon  appel XON
   |   |    |   |          aller à sortie
   |   |    |  fsi
   |   | sinon
   |   |       si BCTS(VOICON(i)) = 1  * CTS bas
   |   |       | alors  appel RTSHAU
   |   |       |    |    si  C = 0     * le CTS est remonté
   |   |       |    |    |   alors  aller à sortie
   |   |       |    |   fsi
   |   |       | sinon  * BCTS(VOICON(i)) = 2
   |   |       |        appel  XOFF
   |   |       |        si  C = 0
   |   |       |        |   alors  aller à sortie
   |  fsi      fsi      fsi
   |   BIRQ(VOICON(i)) := 1       * interdire IRQ
   |   si  SR(VOICON(i),1) = 1     * registre de réception plein
   |   | alors - DBUFR[VOIVON(i),PRT(VOICON(i)] := RHR(VOICON(i))
   |   |       - PRT(VOICON(i)) := PRT(VOICON(i)) + 1
   |   |       - appel BUFPLN
   |   |       - BIRQ(VOICON(i)) := 0
   |   | Sortie:- i := i + 1
fait fsi
fin
```

BUFPLN

```
début
COMPT1(VOICON(i)) := COMPT1(VOICON(i)) + 1
si  COMPT1(VOICON(i)) > LIM
|   alors  RC(VOICON(i)) := 90 H
fsi
fin
```

8.2.- Réception d'un bloc de trois caractères sur une voie basse vitesse: IRQ

```
début
i := 1
tant que VOICON(i) ≠ CR
faire
   | si BIRQ(VOICON(i)) = 0      * POLRBV non interrompue
   | | alors
   | |     si i impair
   | |      | alors
   | |      |    | si ISR(2) = 1
   | |      |    | | alors   * vider la FIFO
   | |      |    | |  pour j de 1 à 3
   | |      |    | |  faire
   | |      |    | |  DBUFR[VOICON(i),PET(VOICON(i)]:=RHR(VOICON(i))
   | |      |    | |  fait
   | |      |    |fsi
   | |      | sinon
   | |      |     si ISR(7) = 1
   | |      |      | alors   * vider la FIFO
   | |      |      |  pour j de 1 à 3
   | |      |      |  faire
   | |      |      |  DBUFR[VOICON(i),PET(VOICON(i)]:=RHR(VOICON(i))
   | |      |      |  fait
   | |      |    fsi
   | |    fsi
   |fsi
   |i := i+1
fait
fin
```

8.3.- Constitution de la trame d'une voie basse vitesse: POLEHV

```
début
i := 1
tant que  VOICON(i) ≠ CR
faire
   |     si  PRQ(VOICON(i)) ≠ PRT(VOICON(i))
   |     |   alors  appel ENFIL
   |    fsi
   |    i := i + 1
fait
fin


ENFIL
début
si PRQ(VOICON i)) > PRT(VOICON(i))
|  alors FILEM(TFILEM) := VOICON(i))
|      |   FILEM(TFILEM + 2) := PRQ(VOICON(i))
|      |  * si contenu de (FBUFR - PRQ) inf. à la long. d'une trame
|      |  si [FBUFR(VOICON(i)) - PRQ(VOICON(i)) + 1] < DIMTRM
|      |  | alors * enfiler cette quantité
|      |  |    |  FILEM(TFILEM+1):=FBUFR(VOICON(i))-PRQ(VOICON(i))+1
|      |  |    |  TFILEM := TFILEM + 3
|      |  |    |  PRQ(VOICON(i)) := DBUFR(VOICON(i))
|      |  | sinon * enfiler une trame
|      |  |       FILEM(TFILEM + 1) := DIMTRM
|      |  |       TFILEM := TFILEM + 3
|      |  |       PRQ(VOICON(i)) := PRQ(VOICON(i)) + DIMTRM
|      | fsi
|  sinon  * PRQ(VOICON(i)) < PRT(VOICON(i))
|       si PRT(VOICON(i)) - PRQ(VOICON(i)) ≥ DIMTRM
|       | alors * enfiler une trame
|       |    |  FILEM(TFILEM) := VOICON(i)
|       |    |  FILEM(TFILEM + 1) := DIMTRM
|       |    |  FILEM(TFILEM + 2) : = PRQ(VOICON(i))
|       |    |  TFILEM := TFILEM + 3
|       |    |  PRQ(VOICON(i)) := PRQ(VOICON(i))+ DIMTRM
|       | sinon * traiter la dernière trame
|       |       appel DERTRA
|       |       si C = 1  *dernière trame non encore enfilée
|       |       |  alors  aller à sortie
|       |      fsi
fsi    fsi
si BFVID = 0   * booléen de file vide
|  alors  BFVID := BFVID + 1
fsi
```

```
FINEM(VOICON(i)) = 0     * émission non encore terminée
si BEMART = 0            * émission a été arrêtée
|  alors
|       si BEOF = 0  * dernière trame entièrement émise
|       |  alors BENFIL := 1     * interdire NMIEMI
|       |       si SR0A(3) = 1   * registre d'émission vide
|       |       |  alors
|       |       |     |  si BEMHV = 2   * émission du numéro de la
|       |       |     |  |  alors            la station secondaire
|       |       |     |  |        CMDRA : = 80 H
|       |       |     |  |        TCTRLA(4) := 1
|       |       |     |  |        DATARA := 5
|       |       |     |  |        CMDRA := C0 H
|       |       |     |  |        BEMHV := 1
|       |       |     |  fsi
|       |       |     |  BEMAR := BEMAR + 1
|       |       |     |  BENFIL := 0
|       |       |     |  si SR0A(3) = 1     * émission entre temps
|       |       |     |  |  alors  appel NMIEMI
|       |       |     |  fsi
|       |       |  sinon  BENFIL := 0
|       |       fsi
|   fsi
fsi
sortie: fin


DERTRA
début
si FINEM(VOICON(i)) ≠ 0       * comptage en cours
|  alors  FINEM(VOICON(i)) := FINEM(VOICON(i)) - 1
|     |     si FINEM(VOICON(i)) = 0    * fin de comptage
|     |     |  alors * enfiler la trame
|     |     |     |  FINEM(TFILEM) := VOICON(i)
|     |     |     |  FINEM(TFILEM+1):= PRT(VOICON(i))-PRQ(VOICON(i))
|     |     |     |  FILEM(TFILEM+2) := PRQ(VOICON(i))
|     |     |     |  PRQ(VOICON(i)):= PRQ(VOICON(i))+FILEM(TFILEM+1)
|     |     |     |  TFILEM := TFILEM + 3
|     |     |     |  C := 0
|     |     |  sinon C := 1    * comptage en cours
|     |   fsi
|  sinon  * début de comptage
|       FILEM(VOICON(i)) := 5FF H
|       C := 1
fsi
fin
```

8.4.- Interruption de la voie composite : NMI

```
début
Charger VECRA     * pour déterminer la cause de l'interruption
si VECRA(1)=0  et  VECRA(2)=1  et  VECRA(3)=1     * Réception
|  alors  appel NMI1
|  sinon
|       si VECRA(1)=0  et  VECRA(2)=0  et  VECRA(3)=1   * émission
|       |  alors  appel NMI2
|       |  sinon
|       |       si VECRA(1)=1  et  VECRA(2)=1  et  VECRA(3)=1
|       |       |  alors  appel NMI3          * condition spéciale
|       |       |  sinon
|       |       |       si VECRA(1)=1 et VECRA(2)=0 et VECRA(3)=1
|       |       |       |  alors appel NMI4 *Interruption Externe
|       |       |      fsi
|       |      fsi
|      fsi
fsi
fin
```

NMI1

```
début
appel NMIREC
si SR0A(7) = 1       * interruption externe entre temps
|  alors
|     |   si BEOF = 1
|     |   |  alors  BEOF := 0
|     |   |     |    CMDRA := 10 H      * réactiver SR0A(7)
|     |   |  sinon
|     |   |       si SPOA(3) = 1      * émission entre temps
|     |   |       |  alors  appel NMIEMI
|     |   |      fsi
|     |  fsi
|  sinon
|       si SR0A(3) = 1            * émission entre temps
|       |  alors  appel NMIEMI
|      fsi
fsi
fin
```

```
NMI2
début
appel NMIEMI
si SR1A(8) = 1       * fin de trame reçue entre temps
|  alors  appel EOF
|  sinon
|        si SR0A(1) = 1       * réception entre temps
|        |  alors  appel NMIREC
fsi     fsi
fin

NMI3
début
si SR1A(8) = 1       * fin de trame reçue
|  alors  appel EOF
|     |    si SR0A(7) = 1      * interruption externe entre temps
|     |    |  alors
|     |    |        si BEOF = 1
|     |    |        |  alors  BEOF := 0
|     |    |        |     |    CMDRA := 10 H      * réactiver SR0A(7)
|     |    |        |  sinon
|     |    |        |        si SR0A(3) = 1  * émission entre temps
|     |    |        |        |  alors  appel NMIEMI
|     |  fsi      fsi      fsi
|  sinon  appel ERREUR
fsi
fin


NMI4
début
si SR0A(7) = 1      * 1er CRC chargé
|  alors  BEOF := 0
|     |    CMDRA := 10 H          * réactiver SR0A(7)
|     |    si SR1A(8) = 1        * fin de trame reçue entre temps
|     |    |  alors  appel EOF
|     |    |  sinon
|     |    |        si SR0A(1) = 1  * si réception entre temps
|     |    |        |  alors  appel NMIREC
|     |  fsi      fsi
|  sinon
|      pour i de 1 à 8
|      faire
|        |  RC(i) := 90 H  *arréter débit des voies basse vitesse
fsi    fait
fin
```

8.5.- Emission d'un caractère sur la voie haute vitesse

NMIEMI

```
début
si BNMIEM = 0      * NMIEMI non interrompue par elle même
|ETQ: alors BNMIEM := BNMIEM + 1
|        |  si BENFIL=0 et BEOF=0 et BPOLRB[FILEM(QFILEM(2))]=0
|        |   | alors *NMIEMI n'inter. pas ENFIL3, trame ent. émise
|        |   |     | si BEMHV = 1     * file d'attente FILEM non vide
|        |   |     |  | alors
|        |   |     |  |     | si BEMHV = 2 *émettre num. de la station
|        |   |     |  |     |  | alors  appel NMIEMI1
|        |   |     |  |     |  | sinon
|        |   |     |  |     |  |      si BEMHV = 1 *émettre num de voie
|        |   |     |  |     |  |       | alors  appel NMIEMI2
|        |   |     |  |     |  |       | sinon *émettre un car de trame
|        |   |     |  |     |  |       |      si FILEM(QFILEM+1) ≠ 0
|        |   |     |  |     |  |       |       | alors appel NMIEMI3
|        |   |     |  |     |  |       |       | sinon appel NMIEMI4
|        |   |     |  |     |  |     fsi    fsi            * trame finie
|        |   |     |  |     | fsi
|        |   |     |  | sinon  BEMART := 0
|        |   |     |  |        CMDRA := 28 H * Réactiver interruption
|        |   |     | fsi                                      d'émission
|        |   |  sinon CMDRA := 28 H *Réactiver interrupt. d'émission
|        |  fsi
|        |   BNMIEM := BNMIEM - 1
|        |   si BNMIEO ≠ 0
|        |    |  alors  aller à ETQ
|        |  fsi
|      sinon  BNMIEM := BNMIEM + 1
|             CMDRA  := 28 H  * Réactiver interruption d'émission
fsi
fin
```

NMIEMI1

```
début
* Emission du numéro de la voie secondaire
si SR0A(3) = 1                  * registre d'émission vide
|  alors  CMDR := 80 H          * m.a.j. du générateur de CRC
|         TCTRLA(4) := 1        * activer le CRC à l'émission
|         DATARA := 5  * envoi du numéro de la station secondaire
|         CMDRA := COH          * terminer la trame par un CRC
|         BEMNV := 1
fsi
fin
```

```
NMIEMI2
début
* Emission du numéro de la voie requise
si SR0A(3) = 1     * registre d'émission vide
|  alors  DATARA := FILEM(QFILEM)
|          BEMHV := 0
fsi
fin
```

```
NMIEMI3

début
* Emission d'un caractère de la trame
si SR0A(3) = 1      * registre d'émission vide
| alors DATARA := (FILEM(QFILEM + 3)
|       FILEM(QFILEM + 2) := FILEM(QFILEM + 2) + 1
|       FILEM(QFILEM + 1) := FILEM(QFILEM + 1) - 1
|       COMPT1(FILEM(QFILEM)) := COMPT1(FILEM(QFILEM)) - 1
|       si COMPT1(FILEM(QFILEM)) ≤ LIM
|        | alors
|        |      si BPLN[FILEM(QFILEM)/2] = 1 * CTS haut après
|        |       | alors RC(FILEM(QFILEM)) := 80 H    buffer plein
|        |       |           * réactiver le débit de la voie basse
|        |      fsi                          vitesse avec CTS bas.
|       fsi
fsi
fin
```

```
NMIEMI4

début
* Fin de trame
pour i de QFILEM à QFILEM+2
faire
  |   FILEM(i) := CR
fait
QFILEM := QFILEM + 3    * m.a.j. de FILEM
si QFILEM = TFILEM
|  alors  BFVID := 0    * file d'attente vide
fsi
BEMHV := 2
BEOF := 1
CMDRA := 28 H  réactiver interruption d'émission
fin
```

8.6.- Réception d'un caractère sur la voie haute vitesse NMIREC

```
début
si  BNMIRE = 0      * NMIREC non interrompue par elle même
|ETQ: alors BNMIRE := BNMIRE + 1
|       |   si BTRAM = 0        * nouvelle trame
|       |    | alors
|       |    |    |  si BNUM = 0  * numéro de la station secondaire
|       |    |    |   | alors  lire le registre de réception
|       |    |    |   |    |    BNUM := 1
|       |    |    |   | sinon * numéro de la voie requise
|       |    |    |   |        NUMDEM := DATARA
|       |    |    |   |        BTRAM := 1
|       |    |    |   |        BNUM := 0
|       |    |    | fsi
|       |    | sinon  * caractére de la trame
|       |    |      si SR0A(1) = 1       * registre plein
|       |    |       | alors DBUFR[PET(NUMDEM)*NUMDEM/2+1] := DATARA
|       |    |       |       PET(NUMDEM) := PET(NUMDEM) + 1
|       |    |     fsi
|       |  fsi
|       |   BNMIRE := BNMIRE - 1
|       |   si BNMIRE ≠ 0
|       |    |  alors  aller à ETQ
|       |  fsi
|     sinon  BNMIRE := BNMIRE + 1
fsi
fin



EOF

début
- lire le registre de réception
- PET(NUMDEM) := PET(NUMDEM) - 1  * retirer le 1er CRC chargé en
- BTRAM := 0                                                mémoire
- CMAKA := 30 H      * réactiver SR 1A (8)
fin
```

8.7.- Emission d'un caractère sur une voie basse vitesse

POLEBV

```
début
i = 1
tant que VOICON(i) ≠ CR
faire
|     si PET(VOICON(i) - PEQ(VOICON(i)) = 0
|     | alors
|     |    |   si PET(VOICON(i)) - PEQ(VOICON(i)) = 0
|     |    |   | alors  aller à sortie
|     |    |   | sinon
|     |    |   |       si PET(VOICON(i)) - PEQ(VOICON(i)) = 1
|     |    |   |       | alors   * traiter dernier caractère
|     |    |   |       |       si  DERCAR(VOICON(i)) ≠ 6
|     |    |   |       |       | alors  aller à sortie
|     |    |   |       |      fsi
|     |    |   |      fsi
|     |    | fsi
|     | sinon
|     |       si PEQ(VOICON(i)) - PET(VOICON(i)) = TAIL - 1
|     |       | alors  * traiter dernier caractère
|     |       |       si DERCAR (VOICON(i)) ≠ 6
|     |       |       | alors  aller à sortie
|     |       |      fsi
|     |      fsi
|    fsi
|   * Emission d'un caractére
|     RHR(VOICON(i)) := DBUFE[VOICON(i), PEQ(VOICON(i))]
|     PEQ(VOICON(i)) := PEQ(VOICON(i)) + 1
|     DERCAR((VOICON(i)) := 0
|Sortie: i := i + 1
fait
fin
```

CHAPITRE 6

CONCLUSIONS ET EXTENSIONS

La simulation, l'étude, la construction et la mise au point d'un multiplexeur temporel statistique à 8 voies, ne relève certes pas d'un travail de routine. Outre la maîtrise des domaines de simulation et de communication, un savoir-faire en matière de microinformatique, matérielle et logicielle, est aussi indispensable. En tout état de cause, la mise en chantier d'un projet de cette envergure réclame le déploiement de gros efforts dans les domaines de compétence évoqués.

Les résultats obtenus par simulation sont équivalents à ceux du produit réel. Ceci a été prouvé dans le paragraphe 2.4.2 du chapitre 4. Les résultats théoriques sont également analogues à ces derniers.

La simulation du produit nous a permis de déterminer la capacité de la mémoire de données requise pour un rendement optimal de la voie composite. Dans le cas où l'on se contente d'un taux de concentration de 100%, 2K de mémoire RAM suffisent, amplement, au stockage des informations échangées. Si par contre, l'on aspire à un taux de concentration supérieur à 100%, il est essentiel de s'assurer, pour une taille mémoire tampon donnée, que les silence sont élevés. En effet, pour des taux de concentration supérieurs à 100%, une optimisation de la charge de la voie composite requiert de très grandes tailles de mémoire tampon, lorsque les silences ne sont pas importants. Et l'on pourra affirmer alors, que plus la mémoire de données sera grande, plus la concentration pourra être élevée. Par simulation, nous avons pu également, tester les cas de stress et leur prévoir les solutions adéquates, puis développer le modèle pour de nouvelles extensions du produit.

En effet, le modèle de base étant élaboré et établi, il s'agira seulement de le compléter, le transformer ou l'adapter, pour l'étude d'éventuelles extensions du projet. Ceci, en fait, avait constitué l'une de nos principales motivations pour la simulation du produit. De nombreuses extensions peuvent être prévues, dont:

— l'augmentation du nombre de voies basse vitesse connectées,
— l'établissement de liaisons locales,
— la retransmission automatique en cas d'erreur,
— la construction d'un véritable réseau de multiplexeurs cascadés,
— etc...

En outre, la simulation du produit, sous sa forme stochastique, sur un système rapide serait vivement recommandée. Elle promet de révéler des résultats fort intéressants.

Enfin, en raison du coût élevé requis pour la simulation de ces systèmes, tout développement théorique de ces derniers serait d'un grand apport pour le domaine de la recherche.

L'objectif de ce projet ne fut donc pas seulement, la construction d'un multiplexeur statistique, mais l'acquisition d'un savoir faire fondamental, pour la construction de toute une gamme de multiplexeurs-concentrateurs.

B I B L I O G R A P H I E

[01] C. Macchi, J.-F. Guilbert et treize co-auteurs, "Téléinformatique, transport et traitement de l'information dans les réseaux et systèmes téléinformatiques et télématiques", collecttion technique et scientifique des télécommunications, Dunod informatique, 1987.

[02] G. Pujolle, D. Seret, D. Dromard, E. Horlait, "Réseaux et télématique. Tome 1", Eyrolles, 1987.

[03] G. Pujolle, "La télématique, réseaux et applications", Eyrolles, 1986.

[04] J.M. Munier, "Introduction à la téléinformatique", Eyrolles, 1986.

[05] S. Krakowiak, "Principe des systèmes d'exploitation des ordinateurs", Dunod informatique, 1987.

[06] CROCUS, "Systèmes d'exploitation des ordinateurs", Dunod informatique, 1981.

[07] F. André, D. Herman, J.-P. Verjus, "Synchronisation de programmes parallèles, expression et mise en oeuvre dans les systèmes centralisés ou distribués", Dunod informatique, 1983.

[08] C. Dardanne, "Le microprocesseur 6809, périphériques et processeur graphique", Eyrolles, 1987.

[09] ———, "Scc2698 Octal Universal Asynchronous Receiver/ Transmitter (Octal-UART)", Signetics, 1987.

[10] ———, "TS68564", Thomson emiconducteurs, 1986.

[11] L. Kleinrock, "Queuing systems, volume I: theory", John Wiley & Sons, 1976.

[12] L. Kleinrock, "Queuing systems, volume II: computer applications", John Wiley & Sons, 1976.

[13] G. Calot, "Cours de statistique descriptive", Dunod décision, 1973.

[14] J. Fourastié, B. Sahler, "Probabilités et statistiques", Dunod, 2ème édition.

[15] M.R. Spiegel, "Probabilités et statistique, cours et problèmes", série Schaum.

[16] F. Riga, "Calculateurs analogiques et hybrides", Techniques de l'ingénieur, traité Informatique, sept. 1989.

[17] A.R. Probst, "Langages de simulation", Techniques de l'ingénieur, traité Informatique, mars 1981.

[18] A.M. Law, W.D. Kelton, "Simulation modeling and analysis", McGraw-Hill, 1982.

[19] P. Bratley, B.L. Fox, L.E. Scharge, "A guide to simulation", Springer-Verlag New York, 1983.

[20] A. Alan B. Pritsker, "Introduction to simulation and SLAM II John Wiley & Sons, 1984.

[21] A. Cernault, "Simulation des systèmes de production", Collection automatisation et production, CEPADUS-EDITIONS, 1988.

[22] M. Pidd, "Computer simulation in management science", Jhon Wiley & sons, 1985.

[23] G.S. Fishman, P.J. Kiviat, "The analysis of simulation-generated time series", Manage. Sci., 13:525-557, 1967.

[24] A.M. Law, "Validation of simulation models, II: comparison of real-world and simulation output data", Univ. Wis. Dept. Ind. Eng. Tech. Rep. 78-15, Madison, 1981.

[25] T.H. Naylor, "Computer simulation experimemts with models of economic systems", Wiley, New York, 1971.

[26] T.H. Naylor, J.M. Finger, "Verification of computer simulation models", Manag. Sci., 14:92-101, 1967.

[27] S.I. Gass, "Evaluation of complex models", Comput. Oper. Res., 4:25-37, 1977.

[28] R.E. Schellenberger, "Criteria for assessing model validity for managerial purpose", Decis. Sci., 5:644-653, 1974.

[29] R.E. shannon, "Systems simulation: the art and science", Prentice-Hall, Englewood Cliffs, N.J., 1975.

[30] R.L. Van Horn, "Validation of simulation results", Manage. Sci., 17:247-258, 1971.

[31] M.A. Crane, A. Lemoine, "An introduction to the regenerative method for simulation analysis", Technical Report No. 86-23, California Analysis Corporation, Palo Alto, CA., Oct. 1976.

[32] E.W. Biles, "Integration-regression search procedure for simulation experimentation", Proceedings, 1974 Winter Simulation Conference, 1974, pp. 491-497.

[33] I.W. Kabak, "Stopping rules for queuing simulations", Operation Research, vol. 16, 1968, pp. 431-437.

[34] S. Karlin, H. Taylor, "A first course in stochastic processes", Academic Press, 1975.

[35] J.P.C. Kleijnen, "Statistical techniques in simulation: part I", Marcel Dekker, 1974.

[36] J.P.C. Kleijnen, "Statistical techniques in simulation: part II", Marcel Dekker, 1975.

[37] A.V. Gafarian, C.J. Ancker, T. Morisaku, "The problem of the initial transient with respect to mean value in digital computer simulation and the evaluation of some proposed solutions", Technical Report No. 77-1, University of Southern California, 1977.

[38] J.R. Wilson, A.A.B. Pritsker, "A survey of research on the simulation startup problem", Simulation, vol. 31, 1978, pp. 55-58.

[39] J.R. Wilson, A.A.B. Pritsker, "A procedure for evaluating

startup policies in simulation experiments", Simulation, vol. 31, 1978, pp. 79-89.

[40] C.H. Sauer, K.M. Chandy, "Computer systems performance modeling", Prentice-Hall, 1981.

[41] B.P. Zeigler, "Theory of modeling and simulation", John Wiley & Sons, 1976.

[42] P.E. Green,Jr & R.W. Lucky, "Computer communications", IEEE Press, 1981.

[43] D.J. Silk, "Queueing model for message switching networks with constant lenght messages", Proc. IEEE, vol. 116, pp. 1822-1826, Nov. 1969.

[44] W.W. Chu, "Buffer behavior for poisson arrivals and multiple synchronous constant outputs", IEEE Trans. Comput., vol. C-19, pp. 530-534, June 1970.

[45] D.G. Maritsas, M.G. Hartley, "Buffer lenght for Erlang input and constant removel rate", IEEE Trans. Comput., vol. C-19, pp. 839-842, Sept. 1970.

[46] W.W. Chu, "Buffer behavior for batch poisson arrivals and single constant output", IEEE Trans. Commun. Technol., vol. C-19, pp. 613-618, Oct. 1970.

[47] H. Rudin, Jr., "Performance of simple multiplexer-concentrators for data communication", IEEE Trans. Commun. Technol., vol. COM-19, pp. 178-187, Apr. 1971.

[48] W.W. Chu, L.C. Liang, "Buffer behavior for mixed input trafic and single constant output", IEEE Trans. Commun., vol. C-20, pp. 230-235, Apr. 1972.

[49] W.W. Chu, "Demultiplexing considerations for statistical multiplexers", IEEE Trans. Commun., vol. C-20, pp. 603-609, June 1972.

[50] D.A. Gall, H.R. Mueller, "Wainting time distribution and buffer overflow in priority queuing systems", IEEE Trans. Commun., vol. C-20, No. 5, Oct. 1972.

[51] D.R. Doll, "Multiplexing and concentration", IEEE Trans. Commun., vol. 60, pp. 1313-1321, Nov. 1972.

[52] Gunnar F. W. Fredriskon, "Buffer behavior for binomial input and constant service", IEEE Trans. Commun., pp. 1862-1866, Nov. 1974.

[53] D. Towsley, J. Wolf, "On the statistical analysis of queue lenghts and waiting times for statistical multiplexers with ARQ retransmissions schemes", IEEE Trans. Commun., vol c-27, No. 4, Apr. 1979.

[54] H. Takagi, "Analysis of polling systems", Cambridge, MA: The M.I.T. Press, 1986.

[55] O.C. Ibe, X. Cheng, "Analysis of polling systems with single message buffers", in Proc. IEEE GLOBECOM '86, Houston, TX, 1986, pp. 939-943.

[56] ZVI Rosberg, "Deterministic routing to buffered channels", IEEE Trans., Commun., vol. C-34, No. 5, May 1986.

[57] O.J. Boxma, W.P. Groenendijk, "Waiting times in discrete-time cyclic-service systems", IEEE Trans. Commun., vol. 36, No. 2, Feb. 1988.

[58] R. Guérin, "Queueing-blocking system with two arrival streams and guard channels", IEEE Trans. Commun., vol. 36, No. 2, Feb. 1988.

[59] H. Akimaru, H. Kuribayashi, T. Inoue, "Approximate evaluation for mixed delay and loss systems with renewal and poisson inputs", IEEE Trans. Commun., vol. 36, No. 7, July 1988

[60] T. Takine, Y. Takahashi, T. Hasegawa, "Exact analysis of asymmetric polling systems with single buffers", IEEE Trans. Commun., vol. 36, pp. 1119-1127, Oct. 1988.

[61] H. Bruneel, "Queuing behavior of statistical multiplexers with correlated inputs", IEEE Trans. Commun., vol. 36, No. 12, Dec. 1988.

[62] A. Ganz, I. Chlamtac, "A linear solution to queueing analysis

of synchronous finite buffer networks", IEEE Trans. Commun., vol. 38, No. 4, Apr. 1990.

[63] S.P. Morgan, C.Y. Lo, "Mean message delays for two packet-FIFO queuing disciplines", IEEE Trans. Commun., vol.38, No. 6, June 1990.

[64] B. Mukherjee, C.K. Kwok, A.C. Lantz, W-H.L. Melody Moh, "Comments on exact analysis of asymmetric polling systems with single buffers", IEEE Trans. Commun., vol. 38, No. 7, July 1990.

[65] W.W. Chu, "A study of asynchronous time division multiplexing for time-sharing computer system", in 1969 Spring Joint Computer Conf., AFIPS Conf. Proc., vol. 35. Montvale, N.J.: AFIPS Press, 1969, pp. 669-678.

[66] K. Seghouani, S. Toumi, "Etude des ressources et de l'rganisation de la clinique de Bou-ismail (W. de Tipaza) à l'aide de la simulation par langage SLAM II", projet de fin d'études, Ecole Natonale Polytechnique, département du Génie Industriel, Juin 1990.

[67] C. Giorgetti, "Mocropower/pascal does real-time control", Digital Equipment Corp., 77 Reed Road, Hudson, Mass. 01749, Nov. 1982.

A N N E X E A

1.- LANGAGES DE SIMULATION CONTINUE

1.1.- CSMP III

CSMP III (Continuous Simulation Modeling Program) est actuellement, le logiciel de simulation continue le plus utilisé. Il met à la disposition de l'utilisateur un certain nombre de blocs fonctionnels semblables à ceux d'une machine analogique. La programmation se fera soit à partir des équations différentielles soit à partir du schéma analogique. Il est à noter que l'utilisateur peut ajouter de nombreux blocs écrits en Fortran.

1.2.- DYNAMO

DYNAMO est un langage de simulation spécialisé, développé pour décrire les modèles selon les principes de la dynamique des systèmes.

Les variables d'état, les variables de flux et les variables auxiliaires du système sont définies. De plus, différents types de délais définissant les temps de réponse des phénomènes peuvent être introduits.

A un instant t, DYNAMO calcule, à l'aide des valeurs des variables d'état et des variables auxiliaires, les valeurs à attribuer au flux durant l'intervalle (t, t+dt). A partir de ces valeurs de flux, il calcule les valeurs des variables d'état et des variables auxiliaires à l'instant (t+dt); puis le temps est incrémenté de dt et la procédure est répétée.

2.- LANGAGES DE SIMULATION DISCRETE

2.1.- GPSS V

GPSS V (General Purpose Simulation System V) est un langage de simulation discrète très populaire à cause de sa simplicité d'utilisation. C'est un langage en blocs. Chaque type de bloc fonctionnel (48 en tout) décrit un ensemble caractéristique de fonctions à simuler et a une représentation graphique. Un modèle en GPSS se présente sous la forme d'une série de blocs reliés. Les entités dynamiques et temporaires qui circulent dans ses blocs sont appelées transactions. Le déplacement des transactions dans le système, le contrôle du temps simulé ainsi que l'accumulation et l'impression des statistiques sont automatiques. Il présente cependant l'inconvénient de ne pouvoir générer des événements à des instants réels (les valeurs réelles sont tronquées).

2.2.- SIMSCRIPT II

SIMSCRIPT est fondé sur les notions d'entités, d'attributs et d'ensembles. L'approche utilisée est celle par événements.

Le langage a été décomposé en niveaux:

— niveau 1 : Instruction de base ;
— niveau 2 : Comparable à Fortran ;
— niveau 3 : Instructions permettant la programmation structurée;
— niveau 4 : Instructions de simulation pour le traitement d'entités, des attributs et des ensembles;
— niveau 5 : Instructions de simulation pour le traitement des événements, pour la collecte de résultats statistiques et la génération de nombres au hasard selon diverses distributions de probabilité.

SIMSCRIPT II a été étendu maintenant, pour permettre également la simulation continue et combinée.

3.- LANGAGES DE SIMULATION COMBINEE

3.1.- GASP IV

GASP IV (General Activity Simulation Program) est un logiciel de simulation combinée écrit en Fortran. Un modèle est décrit par les événements qui peuvent avoir lieu et le mécanisme de leur déclenchement. Les attributs des entités peuvent être discrets ou continus. Des équations différentielles définissent le comportement dynamique des attributs continus.

Le comportement dynamique d'un système est simulé en calculant les valeurs des attributs continus par un algorithme d'intégration et en calculant la valeur des attributs discrets au temps d'occurrence d'un événement.

A N N E X E B

RESULTATS DU SENARIO DU PARAGRAPHE 2.6.3 DU CHAPITRE 4

S L A M I I S U M M A R Y R E P O R T

SIMULATION PROJECT MUX BY TEBIBEL
DATE 14/ 4/1991 RUN NUMBER 1 OF 1

CURRENT TIME .2000E+08
STATISTICAL ARRAYS CLEARED AT TIME .0000E+00

STATISTICS FOR TIME-PERSISTENT VARIABLES

	MEAN VALUE	STANDARD DEVIATION	MINIMUM VALUE	MAXIMUM VALUE	TIME INTERVAL	CURRENT VALUE
TRAME COURANTE	9.511	3.763	.00	11.00	*********	11.00
NB CAR A EMETTRE	10.660	7.521	.00	22.00	*********	8.00
NB TRAM. VOIE 1	.865	.342	.00	1.00	*********	1.00
NB ARRIV. VOIE 1	9599.004	5542.019	.00	19197.00	*********	19197.00

FILE STATISTICS

FILE NUMBER	ASSOC NODE LABEL/TYPE	AVERAGE LENGTH	STANDARD DEVIATION	MAXIMUM LENGTH	CURRENT LENGTH	AVERAGE WAIT TIME
1	AWAIT	.757	.485	2	1	788.734
2		.000	.000	0	0	.000
3		.000	.000	0	0	.000
4		.000	.000	0	0	.000
5		.000	.000	0	0	.000
6		.000	.000	0	0	.000
7		.000	.000	0	0	.000
8		.000	.000	0	0	.000
9		.000	.000	0	0	.000
10	AWAIT	.405	.491	1	1	99.467
11		21.292	1.514	25	20	22183.640
12		.000	.000	0	0	.000
13		.000	.000	0	0	.000
14		.000	.000	0	0	.000
15		.000	.000	0	0	.000
16		.000	.000	0	0	.000
17		.000	.000	0	0	.000
18		.000	.000	0	0	.000
19		.000	.000	0	0	.000

20		2.516	1.181	4	3	2624.251
21	I23 AWAIT	.941	.719	3	1	981.659
22	I23 AWAIT	.000	.000	0	0	.000
23	I23 AWAIT	.000	.000	0	0	.000
24	I23 AWAIT	.000	.000	0	0	.000
25	I23 AWAIT	.000	.000	0	0	.000
26	I23 AWAIT	.000	.000	0	0	.000
27	I23 AWAIT	.000	.000	0	0	.000
28	I23 AWAIT	.000	.000	0	0	.000
29	PREEMPT	.000	.000	1	0	.000
30	PREEMPT	.257	.437	1	1	214.339
31	CALENDAR	4.000	.000	6	4	87.487

REGULAR ACTIVITY STATISTICS

ACTIVITY INDEX/LABEL	AVERAGE UTILIZATION	STANDARD DEVIATION	MAXIMUM UTIL	CURRENT UTIL	ENTITY COUNT
1	.2679	.4428	1	0	19196
2	.0476	.2128	1	0	7927
3	.0104	.1016	1	0	872
4	.0788	.2693	1	0	26251
5	.2406	.4275	1	1	20920
6	.0166	.1279	1	0	3080
7	.1475	.3545	1	0	20917
8	.0000	.0000	1	0	3083
9	.1716	.3771	1	0	19172
10	.0191	.1368	1	0	7951
11	.0000	.0000	0	0	0

RESOURCE STATISTICS

RESOURCE NUMBER	RESOURCE LABEL	CURRENT CAPACITY	AVERAGE UTIL	STANDARD DEVIATION	MAXIMUM UTIL	CURRENT UTIL
1	PROC	1	1.00	.000	1	1

RESOURCE NUMBER	RESOURCE LABEL	CURRENT AVAILABLE	AVERAGE AVAILABLE	MINIMUM AVAILABLE	MAXIMUM AVAILABLE
1	PROC	0	.0000	0	1

GATE STATISTICS

GATE NUMBER	GATE LABEL	CURRENT STATUS	PCT. OF TIME OPEN
1	NONE	CLOSED	.0000

A N N E X E C

RESULTATS DU SENARIO 1 DU PARAGRAPHE 2.6.5 DU CHAPITRE 4

S L A M I I S U M M A R Y R E P O R T

SIMULATION PROJECT MUX BY TEBIBEL

DATE 14/ 4/1991 RUN NUMBER 1 OF 1

CURRENT TIME .2000E+08
STATISTICAL ARRAYS CLEARED AT TIME .7000E+07

STATISTICS FOR TIME-PERSISTENT VARIABLES

	MEAN VALUE	STANDARD DEVIATION	MINIMUM VALUE	MAXIMUM VALUE	TIME INTERVAL	CURRENT VALUE
TRAME COURANTE	12.580	5.342	.00	18.00	*******	11.00
NB CAR A EMETTRE	10.719	7.525	.00	22.00	*******	13.00
NB TRAM. VOIE 1	.143	.350	.00	1.00	*******	1.00
NB TRAM. VOIE 2	.156	.363	.00	1.00	*******	.00
NB TRAM. VOIE 3	.131	.337	.00	1.00	*******	.00
NB TRAM. VOIE 4	.128	.334	.00	1.00	*******	.00
NB TRAM. VOIE 5	.119	.323	.00	1.00	*******	.00
NB TRAM. VOIE 6	.120	.325	.00	1.00	*******	.00
NB TRAM. VOIE 7	.108	.310	.00	1.00	*******	.00
NB TRAM. VOIE 8	.107	.309	.00	1.00	*******	.00
NB ARRIV. VOIE 1	1620.498	450.342	840.00	2401.00	*******	2401.00
NB ARRIV. VOIE 2	1500.498	450.342	720.00	2281.00	*******	2281.00
NB ARRIV. VOIE 3	1380.499	450.342	600.00	2161.00	*******	2161.00
NB ARRIV. VOIE 4	1260.498	450.342	480.00	2041.00	*******	2041.00
NB ARRIV. VOIE 5	1140.504	450.341	360.00	1920.00	*******	1920.00
NB ARRIV. VOIE 6	1020.517	450.339	240.00	1801.00	*******	1801.00
NB ARRIV. VOIE 7	900.520	450.339	120.00	1681.00	*******	1681.00
NB ARRIV. VOIE 8	780.522	450.340	.00	1561.00	*******	1561.00

FILE STATISTICS

FILE NUMBER	ASSOC NODE LABEL/TYPE	AVERAGE LENGTH	STANDARD DEVIATION	MAXIMUM LENGTH	CURRENT LENGTH	AVERAGE WAIT TIME
1	AWAIT	.355	.479	1	1	2956.243
2	AWAIT	.372	.483	1	1	3102.112
3	AWAIT	.395	.489	1	1	3289.587

4		AWAIT	.422	.494	1	1	3517.458
5		AWAIT	.447	.497	1	0	3721.197
6		AWAIT	.465	.499	1	1	3876.647
7		AWAIT	.497	.500	1	1	4136.639
8		AWAIT	.514	.500	1	1	4282.818
9			.000	.000	0	0	.000
10		AWAIT	.405	.491	1	1	99.443
11			13.121	5.739	23	15	*********
12			13.180	5.693	23	14	*********
13			12.684	5.743	23	4	*********
14			12.727	5.720	23	16	*********
15			12.227	5.758	23	6	*********
16			12.280	5.732	23	18	*********
17			11.732	5.746	22	8	97141.120
18			11.827	5.734	22	20	98556.110
19			.143	.350	1	0	4670.194
20			2.527	1.175	4	3	2636.539
21	I23	AWAIT	6.960	6.409	20	5	57888.760
22	I23	AWAIT	6.278	6.123	20	3	52251.760
23	I23	AWAIT	7.209	6.362	20	18	59464.530
24	I23	AWAIT	6.681	6.484	20	0	55603.130
25	I23	AWAIT	7.361	6.404	20	15	60875.910
26	I23	AWAIT	6.773	6.613	20	0	56367.790
27	I23	AWAIT	7.073	6.335	20	12	58605.970
28	I23	AWAIT	6.760	6.574	20	0	57068.000
29		PREEMPT	.000	.000	0	0	.000
30		PREEMPT	.257	.437	1	1	214.312
31		CALENDAR	11.000	.000	12	11	260.519

REGULAR ACTIVITY STATISTICS

ACTIVITY INDEX/LABEL	AVERAGE UTILIZATION	STANDARD DEVIATION	MAXIMUM UTIL	CURRENT UTIL	ENTITY COUNT
1	.2680	.4429	1	0	12480
2	.0475	.2128	1	0	5150
3	.0104	.1016	1	0	567
4	.0788	.2694	1	0	17065
5	.2405	.4274	1	1	13595
6	.0167	.1280	1	0	2005
7	.1474	.3545	1	0	13592
8	.0000	.0000	1	0	2008
9	.1716	.3770	1	0	12453
10	.0191	.1370	1	0	5179
11	.0000	.0000	0	0	0

A N N E X E D

RESULTATS DU SENARIO 2 DU PARAGRAPHE 2.6.5 DU CHAPITRE 4

S L A M I I S U M M A R Y R E P O R T

SIMULATION PROJECT MUX BY TEBIBEL

DATE 14/ 4/1991 RUN NUMBER 1 OF 1

CURRENT TIME .2000E+08
STATISTICAL ARRAYS CLEARED AT TIME .0000E+00

STATISTICS FOR TIME-PERSISTENT VARIABLES

	MEAN VALUE	STANDARD DEVIATION	MINIMUM VALUE	MAXIMUM VALUE	TIME INTERVAL	CURRENT VALUE
TRAME COURANTE	.985	3.696	.00	18.00	******	.00
NB CAR A EMETTRE	.940	3.844	.00	22.00	******	.00
NB TRAM. VOIE 1	.150	.610	.00	4.00	******	.00
NB TRAM. VOIE 2	.162	.656	.00	4.00	******	.00
NB TRAM. VOIE 3	.126	.556	.00	4.00	******	.00
NB TRAM. VOIE 4	.134	.589	.00	4.00	******	.00
NB TRAM. VOIE 5	.146	.627	.00	4.00	******	.00
NB TRAM. VOIE 6	.136	.571	.00	4.00	******	.00
NB TRAM. VOIE 7	.134	.571	.00	4.00	******	.00
NB TRAM. VOIE 8	.146	.607	.00	4.00	******	.00
NB ARRIV. VOIE 1	2400.491	1385.622	.00	4800.00	******	4800.00
NB ARRIV. VOIE 2	2400.491	1385.622	.00	4800.00	******	4800.00
NB ARRIV. VOIE 3	2400.491	1385.622	.00	4800.00	******	4800.00
NB ARRIV. VOIE 4	2400.491	1385.622	.00	4800.00	******	4800.00
NB ARRIV. VOIE 5	2400.491	1385.622	.00	4800.00	******	4800.00
NB ARRIV. VOIE 6	2400.491	1385.622	.00	4800.00	******	4800.00
NB ARRIV. VOIE 7	2400.491	1385.622	.00	4800.00	******	4800.00
NB ARRIV. VOIE 8	2400.491	1385.622	.00	4800.00	******	4800.00
NB DEBOR. VOIE 1	2194.102	1378.029	.00	4589.00	******	4589.00
NB DEBOR. VOIE 2	2193.186	1377.891	.00	4588.00	******	4588.00
NB DEBOR. VOIE 3	2215.221	1379.411	.00	4611.00	******	4611.00
NB DEBOR. VOIE 4	2214.260	1379.353	.00	4610.00	******	4610.00
NB DEBOR. VOIE 5	2214.294	1379.302	.00	4610.00	******	4610.00
NB DEBOR. VOIE 6	2193.140	1377.967	.00	4588.00	******	4588.00
NB DEBOR. VOIE 7	2213.317	1379.266	.00	4609.00	******	4609.00
NB DEBOR. VOIE 8	2193.130	1377.983	.00	4588.00	******	4588.00

FILE STATISTICS

FILE NUMBER	ASSOC NODE LABEL/TYPE	AVERAGE LENGTH	STANDARD DEVIATION	MAXIMUM LENGTH	CURRENT LENGTH	AVERAGE WAIT TIME
1	AWAIT	.063	.349	3	0	6003.414
2	AWAIT	.066	.360	3	0	6272.040
3	AWAIT	.059	.339	3	0	6212.721
4	AWAIT	.060	.343	3	0	6290.323
5	AWAIT	.059	.339	3	0	6242.112
6	AWAIT	.065	.355	3	0	6159.960
7	AWAIT	.059	.340	3	0	6218.082
8	AWAIT	.064	.349	3	0	6057.654
9		.000	.000	0	0	.000
10	AWAIT	.151	.358	1	0	13.839
11		16.145	12.862	101	13	*********
12		17.341	13.664	102	14	*********
13		15.605	11.655	101	13	*********
14		16.745	12.254	102	14	*********
15		17.020	13.142	102	14	*********
16		16.729	11.755	102	14	*********
17		17.668	11.708	103	15	*********
18		16.961	12.535	102	14	*********
19		1.065	4.489	30	0	*********
20		.198	.754	4	0	2640.770
21	23 AWAIT	1.196	5.287	42	0	*********
22	I23 AWAIT	1.047	4.684	38	0	*********
23	I23 AWAIT	.809	3.836	35	0	91919.900
24	I23 AWAIT	.713	3.337	29	0	81065.890
25	I23 AWAIT	.641	3.013	25	0	72860.180
26	I23 AWAIT	1.058	4.564	35	0	*********
27	I23 AWAIT	.752	3.452	29	0	85489.320
28	I23 AWAIT	1.152	5.004	38	0	*********
29	PREEMPT	.000	.000	1	0	.000
30	PREEMPT	.140	.347	1	0	116.296
31	CALENDAR	11.000	.000	13	11	146.870

REGULAR ACTIVITY STATISTICS

ACTIVITY INDEX/LABEL	AVERAGE UTILIZATION	STANDARD DEVIATION	MAXIMUM UTIL	CURRENT UTIL	ENTITY COUNT
1	.0090	.0944	1	0	644
2	.4309	.4952	1	1	71811
3	.0008	.0285	1	0	68
4	.2181	.4130	1	0	72708
5	.0188	.1357	1	0	1632
6	.1208	.3259	1	0	22369
7	.0115	.1066	1	0	1632
8	.0000	.0000	1	0	22369
9	.0134	.1149	1	0	1496
10	.1711	.3766	1	0	71280
11	.0057	.0752	1	0	321

ANNEXE E

RESULTATS DU SENARIO 3 DU PARAGRAPHE 2.6.5 DU CHAPITRE 4

S L A M I I S U M M A R Y R E P O R T

SIMULATION PROJECT MUX BY TEBIBEL

DATE 14/ 4/1991 RUN NUMBER 1 OF 1

CURRENT TIME .2000E+08
STATISTICAL ARRAYS CLEARED AT TIME .0000E+00

STATISTICS FOR TIME-PERSISTENT VARIABLES

	MEAN VALUE	STANDARD DEVIATION	MINIMUM VALUE	MAXIMUM VALUE	TIME INTERVAL	CURRENT VALUE
TRAME COURANTE	6.487	8.109	.00	18.00	*******	18.00
NB CAR A EMETTRE	4.963	7.421	.00	22.00	*******	12.00
NB TRAM. VOIE 1	.193	.699	.00	4.00	*******	.00
NB TRAM. VOIE 2	.186	.676	.00	4.00	*******	.00
NB TRAM. VOIE 3	.182	.663	.00	4.00	*******	.00
NB TRAM. VOIE 4	.193	.688	.00	4.00	*******	.00
NB TRAM. VOIE 5	.186	.672	.00	4.00	*******	.00
NB TRAM. VOIE 6	.176	.667	.00	4.00	*******	.00
NB TRAM. VOIE 7	.194	.395	.00	1.00	*******	.00
NB TRAM. VOIE 8	.107	.309	.00	1.00	*******	1.00
NB ARRIV. VOIE 1	2400.490	1385.621	.00	4800.00	*******	4800.00
NB ARRIV. VOIE 2	2400.486	1385.621	.00	4800.00	*******	4800.00
NB ARRIV. VOIE 3	2400.443	1385.622	.00	4800.00	*******	4800.00
NB ARRIV. VOIE 4	2400.010	1385.622	.00	4800.00	*******	4800.00
NB ARRIV. VOIE 5	2395.700	1385.613	.00	4796.00	*******	4796.00
NB ARRIV. VOIE 6	2352.682	1385.210	.00	4752.00	*******	4752.00
NB ARRIV. VOIE 7	1944.419	1348.958	.00	4320.00	*******	4320.00
NB ARRIV. VOIE 8	600.217	774.758	.00	2400.00	*******	2400.00
NB DEBOR. VOIE 1	2007.939	1358.417	.00	4390.00	*******	4390.00
NB DEBOR. VOIE 2	2028.108	1361.127	.00	4412.00	*******	4412.00
NB DEBOR. VOIE 3	2047.462	1363.523	.00	4433.00	*******	4433.00
NB DEBOR. VOIE 4	2048.868	1363.768	.00	4435.00	*******	4435.00
NB DEBOR. VOIE 5	2065.236	1365.746	.00	4453.00	*******	4453.00
NB DEBOR. VOIE 6	2065.051	1365.724	.00	4452.00	*******	4452.00
NB DEBOR. VOIE 7	.000	.000	.00	.00	*******	.00
NB DEBOR. VOIE 8	.000	.000	.00	.00	*******	.00

FILE STATISTICS

FILE NUMBER	ASSOC LABEL	NODE TYPE	AVERAGE LENGTH	STANDARD DEVIATION	MAXIMUM LENGTH	CURRENT LENGTH	AVERAGE WAIT TIME
1		AWAIT	.121	.464	3	0	5921.625
2		AWAIT	.116	.455	3	0	5967.310
3		AWAIT	.107	.437	3	0	5833.707
4		AWAIT	.102	.422	3	0	5576.957
5		AWAIT	.102	.430	3	0	5956.600
6		AWAIT	.093	.412	3	0	6178.872
7		AWAIT	.391	.490	2	0	1810.577
8		AWAIT	.251	.438	2	1	2108.432
9			.000	.000	0	0	.000
10		AWAIT	.256	.436	1	1	31.482
11			17.830	14.182	102	14	*********
12			17.681	13.737	102	14	*********
13			18.553	13.362	103	15	*********
14			16.942	14.180	101	13	*********
15			16.780	13.849	101	13	*********
16			17.391	13.657	102	14	*********
17			11.933	6.305	23	8	55245.050
18			6.629	7.548	23	13	55263.670
19			1.021	3.698	22	0	*********
20			1.155	1.488	4	3	2627.914
21	I23	AWAIT	5.419	16.955	93	0	*********
22	I23	AWAIT	4.111	13.317	69	0	*********
23	I23	AWAIT	3.893	12.703	69	0	*********
24	I23	AWAIT	4.484	14.622	75	0	*********
25	I23	AWAIT	3.918	13.133	73	0	*********
26	I23	AWAIT	2.771	10.010	59	0	*********
27	I23	AWAIT	5.895	5.946	18	14	27342.240
28	I23	AWAIT	3.187	5.323	18	6	26748.940
29		PREEMPT	.000	.000	1	0	.000
30		PREEMPT	.188	.391	1	1	156.737
31		CALENDAR	11.000	.000	12	11	175.131

REGULAR ACTIVITY STATISTICS

ACTIVITY INDEX/LABEL	AVERAGE UTILIZATION	STANDARD DEVIATION	MAXIMUM UTIL	CURRENT UTIL	ENTITY COUNT
1	.1081	.3106	1	0	7749
2	.2761	.4471	1	0	46013
3	.0048	.0690	1	0	400
4	.1612	.3677	1	0	53736
5	.1103	.3132	1	1	9588
6	.0778	.2679	1	0	14412
7	.0676	.2510	1	0	9585
8	.0000	.0000	1	0	14415
9	.0785	.2689	1	0	8765
10	.1089	.3115	1	0	45371
11	.0067	.0818	1	0	381

A N N E X E F

RESULTATS DU SENARIO 4 DU PARAGRAPHE 2.6.5 DU CHAPITRE 4

S L A M I I S U M M A R Y R E P O R T

SIMULATION PROJECT MUX BY TEBIBEL

DATE 14/ 4/1991 RUN NUMBER 1 OF 1

CURRENT TIME .2000E+08
STATISTICAL ARRAYS CLEARED AT TIME .0000E+00

STATISTICS FOR TIME-PERSISTENT VARIABLES

	MEAN VALUE	STANDARD DEVIATION	MINIMUM VALUE	MAXIMUM VALUE	TIME INTERVAL	CURRENT VALUE
TRAME COURANTE	6.975	6.765	.00	18.00	*******	.00
NB CAR A EMETTRE	6.606	7.897	.00	22.00	*******	.00
NB TRAM. VOIE 1	.418	.811	.00	4.00	*******	.00
NB TRAM. VOIE 2	.396	.808	.00	4.00	*******	.00
NB TRAM. VOIE 3	.368	.809	.00	4.00	*******	.00
NB TRAM. VOIE 4	.333	.790	.00	4.00	*******	.00
NB TRAM. VOIE 5	.309	.784	.00	4.00	*******	.00
NB TRAM. VOIE 6	.125	.590	.00	4.00	*******	.00
NB TRAM. VOIE 7	.054	.225	.00	1.00	*******	.00
NB TRAM. VOIE 8	.027	.161	.00	1.00	*******	.00
NB ARRIV. VOIE 1	2400.490	1385.621	.00	4800.00	*******	4800.00
NB ARRIV. VOIE 2	1837.908	1329.938	.00	4200.00	*******	4200.00
NB ARRIV. VOIE 3	1350.346	1190.695	.00	3600.00	*******	3600.00
NB ARRIV. VOIE 4	937.767	998.178	.00	3000.00	*******	3000.00
NB ARRIV. VOIE 5	600.217	774.758	.00	2400.00	*******	2400.00
NB ARRIV. VOIE 6	337.676	539.717	.00	1801.00	*******	1801.00
NB ARRIV. VOIE 7	150.128	312.443	.00	1201.00	*******	1201.00
NB ARRIV. VOIE 8	37.569	116.752	.00	601.00	*******	601.00
NB DEBOR. VOIE 1	272.841	468.981	.00	1618.00	*******	1618.00
NB DEBOR. VOIE 2	278.603	475.542	.00	1635.00	*******	1635.00
NB DEBOR. VOIE 3	276.224	472.839	.00	1628.00	*******	1628.00
NB DEBOR. VOIE 4	274.183	470.515	.00	1622.00	*******	1622.00
NB DEBOR. VOIE 5	279.967	477.087	.00	1639.00	*******	1639.00
NB DEBOR. VOIE 6	277.938	474.788	.00	1634.00	*******	1634.00
NB DEBOR. VOIE 7	.000	.000	.00	.00	*******	.00
NB DEBOR. VOIE 8	.000	.000	.00	.00	*******	.00

FILE STATISTICS

FILE NUMBER	ASSOC NODE LABEL/TYPE	AVERAGE LENGTH	STANDARD DEVIATION	MAXIMUM LENGTH	CURRENT LENGTH	AVERAGE WAIT TIME
1	AWAIT	.608	.752	3	0	3820.196
2	AWAIT	.558	.753	3	0	4347.484
3	AWAIT	.519	.776	3	0	5262.174
4	AWAIT	.392	.740	3	0	5684.662
5	AWAIT	.225	.605	3	0	5915.803
6	AWAIT	.049	.300	3	0	5904.298
7	AWAIT	.107	.309	2	1	1777.334
8	AWAIT	.064	.246	2	1	2113.283
9		.000	.000	0	0	.000
10	AWAIT	.298	.457	1	0	42.107
11		19.766	16.322	102	14	*********
12		17.839	17.490	101	13	*********
13		16.434	18.444	102	14	*********
14		14.529	18.823	102	14	*********
15		12.652	19.318	101	13	*********
16		7.471	15.313	101	13	*********
17		3.307	6.284	23	12	55116.860
18		1.649	4.728	23	6	54959.350
19		1.500	3.983	22	0	*********
20		1.543	1.536	4	0	2630.958
21	I23 AWAIT	57.025	90.282	307	0	*********
22	I23 AWAIT	55.189	89.398	306	0	*********
23	I23 AWAIT	55.125	90.734	306	0	*********
24	I23 AWAIT	53.615	91.011	305	0	*********
25	I23 AWAIT	21.066	48.801	192	0	*********
26	I23 AWAIT	.690	3.584	32	0	89666.610
27	I23 AWAIT	1.588	4.035	18	9	26735.160
28	I23 AWAIT	.780	2.973	18	16	26277.250
29	PREEMPT	.000	.000	1	0	.000
30	PREEMPT	.208	.406	1	0	173.024
31	CALENDAR	11.000	.000	12	11	192.937

REGULAR ACTIVITY STATISTICS

ACTIVITY INDEX/LABEL	AVERAGE UTILIZATION	STANDARD DEVIATION	MAXIMUM UTIL	CURRENT UTIL	ENTITY COUNT
1	.1275	.3335	1	1	9137
2	.2228	.4161	1	0	37125
3	.0064	.0796	1	0	533
4	.1399	.3468	1	0	46619
5	.1471	.3542	1	0	12792
6	.0605	.2385	1	0	11209
7	.0902	.2864	1	0	12792
8	.0000	.0000	1	0	11209
9	.1048	.3062	1	0	11701
10	.0851	.2790	1	0	35451
11	.0159	.1249	1	0	896