Étude de faisabilité d'un micro-contrôleur de très haute sécurité

Gilles Chaumontet

THESE

présentée par

Gilles CHAUMONTET

pour obtenir le titre de DOCTEUR

de L'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE
(arrêté ministériel du 23 novembre 1988)

option micro-électronique

ETUDE DE FAISABILITE D'UN MICRO-CONTROLEUR DE TRES HAUTE SECURITE

date de soutenance : 26 octobre 1990

composition du jury : M VERDONE Max président

MM. COURTOIS Bernard
GREINER Alain rapporteur
GRUERE Yves
NICOLAIDIS Mihalis
TEIXEIRA J.Paulo rapporteur

Thèse préparée au sein du laboratoire TIM3/INPG

JOSELEAU Jean Paul	Biochimie
KAHANE André, détaché	Physique
KAHANE Josette	Physique
KRAKOWIAK Sacha	Mathématiques Appliquées
LAJZEROWICZ Jeanine	Physique
LAJZEROWICZ Joseph	Physique
LAURENT Pierre-Jean	Mathématiques Appliquées
LEBRETON Alain	Mathématiques Appliquées
DE LEIRIS Joël	Biologie
LHOMME Jean	Chimie
LLIBOUTRY Louis	Géophysique
LOISEAUX Jean-Marie	Sciences Nucléaires I.S.N.
LONGEQUEUE Nicole	Physique
LUNA Domingo	Mathématiques Pures
MACHE Régis	Physiologie Végétale
MASCLE Georges	Géologie
MAYNARD Roger	Physique du Solide
OMONT Alain	Astrophysique
OZENDA Paul	Botanique (Biologie Végétale)
PANNETIER Jean	Chimie
PAYAN Jean-Jacques	Mathématiques Pures
PEBAY-PEYROULA Jean-Claude	Physique
PERRIER Guy	Géophysique
PIERRE Jean Louis	Chimie Organique
RENARD Michel	Thermodynamique
RIEDTMANN Christine	Mathématiques
RINAUDO Marguerite	Chimie CERMAV
ROSSI André	Biologie
SAXOD Raymond	Biologie Animale
SENGEL Philippe	Biologie Animale
SERGERAERT Francis	Mathématiques Pures
SOUCHIER Bernard	Biologie
SOUTIF Michel	Physique
STUTZ Pierre	Mécanique
TRILLING Laurent	Mathématiques Appliquées
VAN CUTSEM Bernard	Mathématiques Appliquées
VIALON Pierre	Géologie

PROFESSEURS de 2ème Classe

ARMAND Gilbert	Géographie
ATTANE Pierre	Mécanique
BARET Paul	Chimie
BERTIN José	Mathématiques
BLANCHI J.Pierre	STAPS
BLOCK Marc	Biologie
BLUM Jacques	Mathématiques Appliquées
BOITET Christian	Mathématiques Appliquées
BORNAREL Jean	Physique
BORRIONE Dominique	Automatique informatique
BOUVET Jean	Biologie
BROSSARD Jean	Mathématiques
BRUANDET J.François	Physique
BRUGAL Gérard	Biologie
BRUN Gilbert	Biologie
CASTAING Bernard	Physique
CERFF Rudiger	Biologie
CHIARAMELLA Yves	Mathématiques Appliquées
CHOLLET Jean Pierre	Mécanique
COLOMBEAU Jean François	Mathématiques (ENSL)
COURT Jean	Chimie
CUNIN Pierre Yves	Informatique
DAVID Jean	Géographie

DHOUAILLY Danielle	Biologie
DUFRESNOY Alain	Mathématiques Pures
GASPARD François	Physique
GIDON Maurice	Géologie
GIGNOUX Claude	Sciences Nucléaires
GILLARD Roland	Mathématiques Pures
GIORNI Alain	Sciences Nucléaires
GONZALEZ SPRINBERG Gérardo	Mathématiques Pures
GUIGO Maryse	Géographie
GUMUCHAIN Hervé	Géographie
HACQUES Gérard	Mathématiques Appliquées
HERBIN Jacky	Géographie
HERAULT Jeanny	Physique
HERINO Roland	Physique
JARDON Pierre	Chimie
KERCKHOVE Claude	Géologie
MANDARON Paul	Biologie
MARTINEZ Francis	Mathématiques Appliquées
MOREL Alain	Géographie
NEMOZ Alain	Thermodymamique CNRS - CRTBT
NGUYEN HUY Xuong	Informatique
OUDET Bruno	Mathématiques Appliquées
PAUTOU Guy	Biologie
PECHER Arnaud	Géologie
PELMONT Jean	Biochimie
PELLETIER Guy	Astrophysique
PERRIN Claude	Sciences Nucléaires I.S.N.
PIBOULE Michel	Géologie
RAYNAUD Hervé	Mathématiques Appliquées
REGNARD Jean René	Physique
RICHARD Jean-Marc	Physique
RIEDTMANN Christine	Mathématiques Pures
ROBERT Danielle	Chimie
ROBERT Gilles	Mathématiques Pures
ROBERT Jean-Bernard	Chimie Physique
SARROT-REYNAULD Jean	Géologie
SAYETAT Françoise	Physique
SERVE Denis	Chimie
STOECKEL Frédéric	Physique
SCHOLL Pierre-Claude	Mathématiques Appliquées
SUBRA Robert	Chimie
VALLADE Marcel	Physique
VIDAL Michel	Chimie Organique
VINCENT Gilbert	Physique
VIVIAN Robert	Géographie
VOTTERO Philippe	Chimie

MEMBRES DU CORPS ENSEIGNANT DE L' IUT 1

PROFESSEURS de 1ère Classe

BUISSON Roger	Physique IUT 1
CHEHIKIAN Alain	E.E.A. I.U.T.1
DODU Jacques	Mécanique Appliquée IUT 1
NEGRE Robert	Génie Civil IUT 1
NOUGARET Marcel	Automatique IUT 1
PERARD Jacques	EEA. IUT 1

PROFESSEURS de 2ème classe

BEE Marc	Physique IUT 1
BOUTHINON Michel	EEA. IUT 1
CHAMBON René	Génie Mécanique IUT 1
CHENAVAS Jean	Physique IUT 1

CHILO Jean	Physique IUT 1
CHOUTEAU Gérard	Physique IUT 1
CONTE René	Physique IUT 1
FOSTER Panayotis	Chimie IUT 1
GOSSE Jean Pierre	EEA.IUT 1
GROS Yves	Physique IUT 1
HAMAR Roger	Chimie IUT 1
KUHN Gérard, (Détaché)	Physique IUT 1
LEVIEL Jean Louis	Physique IUT 1
MAZUER Jean	Physique IUT 1
MICHOULIER Jean	Physique IUT 1
MONLLOR Christian	EEA.IUT 1
PERRAUD Robert	Chimie IUT 1
PIERRE Gérard	Chimie IUT 1
TERRIEZ Jean-Michel	Génie Mécanique IUT 1
TOUZAIN Philippe	Chimie IUT 1
TURGEMAN Sylvain	Génie civil
VINCENDON Marc	Chimie IUT 1
ZIGONE Michel	Physique IUT 1

PROFESSEURS DE PHARMACIE

AGNIUS-DELORD Claudine	Physique	Faculté La Tronche
ALARY Josette	Chimie Analytique	Faculté La Tronche
BERIEL Hélène	Physiologie et Pharmacologie	Faculté La Tronche
CUSSAC Max	Chimie Therapeutique	Faculté La Tronche
DEMENGE Pierre	Pharmacodynamie	Faculté La Tronche
FAVIER Alain	Biochimie	C.H.R.G.
JEANNIN Charles	Pharmacie Galénique	Faculté Meylan
LATURAZE Jean	Biochimie	Faculté La Tronche
LUU DUC Cuong	Chimie Générale	Faculté La Tronche
MARIOTTE Anne-Marie	Pharmacognosie	Faculté La Tronche
MARZIN Daniel	Toxicologie	Faculté Meylan
RENAUDET Jacqueline	Bactériologie	Faculté La Tronche
ROCHAT Jacques	Hygiéne et Hydrologie	Faculté La Tronche
SEIGLE-MURANDI Françoise	Botanique et Cryptogamie	Faculté Meylan
VERAIN Alice	Pharmacie Galénique	Faculté Meylan

MEMBRES DU CORPS ENSEIGNANT DE MEDECINE

PROFESSEURS CLASSE EXEPTIONNELLE ET 1ère CLASSE

AMBLARD Pierre	Dermatologie	C.H.R.G.
AMBROISE-THOMAS Pierre	Parasitologie	C.H.R.G.
BEAUDOING André	Pédiatrie-Puericulture	C.H.R.G.
BEZEZ Henri	Orthopédie-Traumatologie	Hopital SUD
BONNET Jean-Louis	Ophtalmologie	C.H.R.G.
BOUCHET Yves	Anatomie	Faculté La Merci
	Chirurgie Générale et Digestive	C.H.R.G.
BUTEL Jean	Orthopédie-Traumatologie	C.H.R.G.
CHAMBAZ Edmond	Biochimie	C.H.R.G.
CHAMPETIER Jean	Anatomie-Topographique et Appliquée	C.H.R.G.
CHARACHON Robert	O.R.L.	C.H.R.G.
COLOMB Maurice	Immunologie	Hopital sud
COUDERC Pierre	Anatomie-Pathologique	C.H.R.G.
DELORMAS Pierre	Pneumophtisiologie	C.H.R.G.
DENIS Bernard	Cardiologie	C.H.R.G.
GAVEND Michel	Pharmacologie	Faculté La Merci

UNIVERSITE Joseph FOURIER (GRENOBLE I)

Président de l'Université :
M. NEMOZ Alain

Année Universitaire 1988 - 1989

MEMBRES DU CORPS ENSEIGNANT DE SCIENCES ET DE GEOGRAPHIE

PROFESSEURS DE 1ère Classe

ADIBA Michel	Informatique
ANTOINE Pierre	Géologie I.R.I.G.M.
ARNAUD Paul	Chimie Organique
ARVIEU Robert	Physique Nucléaire I.S.N.
AUBERT Guy	Physique C.N.R.S
AURIAULT Jean-Louis	Mécanique
AYANT Yves	Physique Approfondie
BARBIER Marie-Jeanne	Electrochimie
BARJON Robert	Physique Nucléaire ISN
BARNOUD Fernand	Biochimie Macromoléculaire Végétale
BARRA Jean-René	Statistiques-Mathématiques Appliquées
BECKER Pierre	Physique
BEGUIN Claude	Chimie Organique
BELORISKY Elie	Physique
BENZAKEN Claude	Mathématiques Pures
BERARD Pierre	Mathématiques Pures
BERNARD Alain	Mathématiques Pures
BERTRANDIAS Françoise	Mathématiques Pures
BERTRANDIAS Jean-Paul	Mathématiques Pures
BILLET Jean	Géographie
BOELHER Jean-Paul	Mécanique
BRAVARD Yves	Géographie
CARLIER Georges	Biologie Végétale
CASTAING Bernard	Physique
CAUQUIS Georges	Chimie Organique
CHARDON Michel	Géographie
CHIBON Pierre	Biologie Animale
COHEN ADDAD Jean-Pierre	Physique
COLIN DE VERDIERE Yves	Mathématiques Pures
CYROT Michel	Physique du Solide
DEBELMAS Jacques	Géologie Générale
DEGRANGE Charles	Zoologie
DEMAILLY Jean-Pierre	Mathématiques Pures
DENEUVILLE Alain	Physique
DEPORTES Charles	Chimie Minérale
DOLIQUE Jean-Michel	Physique des Plasmas
DOUCE Roland	Physiologie Végétale
DUCROS Pierre	Cristallographie
FINKE Gerde	Informatique
GAGNAIRE Didier	Chimie Physique
GAUTRON René	Chimie
GENIES Eugène	Chimie
GERMAIN Jean-Pierre	Mécanique,
GIDON Maurice	Géologie
GUITTON Jacques	Chimie
HICTER Pierre	Chimie
IDELMAN Simon	Physiologie Animale
JANIN Bernard	Géographie
JOLY Jean René	Mathématiques Pures

HOLLARD Daniel	Hématologie	C.H.R.G.
LATREILLE René	Chirurgie Thoracique et Cardiovasculaire	C.H.R.G.
LE NOC Pierre	Bactériologie-Virologie	C.H.R.G.
MALINAS Yves	Gynécologie et Obstétrique	C.H.R.G.
MALLION Jean-Michel	Médecine du Travail	C.H.R.G.
MICOUD Max	Clinique Médicale et Maladies Infectieuses	C.H.R.G.
MOURIQUAND Claude	Histologie	Faculté La Merci
PARAMELLE Bernard	Pneumologie	C.H.R.G.
PERRET Jean	Neurologie	C.H.R.G.
RACHAIL Michel	Hépato-Gastro-Entérologie	C.H.R.G.
DE ROUGEMONT Jacques	Neurochirurgie	C.H.R.G.
SARRAZIN Roger	Clinique Chirurgicale	C.H.R.G.
STIEGLITZ Paul	Anestésiologie	C.H.R.G.
TANCHE Maurice	Physiologie	Faculté La Merci
VIGNAIS Pierre	Biochimie	Faculté La Merci

PROFESSEURS 2ème CLASSE

BACHELOT Yvan	Endocrinologie	C.H.R.G.
BARGE Michel	Neurochirurgie	C.H.R.G.
BENABID Alim Louis	Biophysique	Faculté La Merci
BENSA Jean-Claude	Immunologie	Hopital Sud
BERNARD Pierre	Gynécologie-Obstétrique	C.H.R.G.
BESSARD Germain	Pharmacologie	ABIDJAN
BOLLA Michel	Radiothérapie	C.H.R.G.
BOST Michel	Pédiatrie	C.H.R.G.
BOUCHARLAT Jacques	Psychiatrie Adultes	Hopital Sud
BRAMBILLA Christian	Pneumologie	C.H.R.G.
CHIROSSEL Jean-Paul	Anatomie-Neurochirurgie	C.H.R.G.
COMET Michel	Biophysique	Faculté La Merci
CONTAMIN Charles	Chirurgie Thoracique et Cardiovasculaire	C.H.R.G.
CORDONNIER Daniel	Néphrologie	C.H.R.G.
COULOMB Max	Radiologie	C.H.R.G.
CROUZET Guy	Radiologie	C.H.R.G.
DEBRU Jean-Luc	Médecine Interne et Toxicologie	C.H.R.G.
DEMONGEOT Jacques	Biostatistiques et Informatique Médicale	Faculté La Merci
DUPRE Alain	Chirurgie Générale	C.H.R.G.
DYON Jean-François	Chirurgie Infantile	C.H.R.G.
ETERRADOSSI Jacqueline	Physiologie	Faculté La Merci
FAURE Claude	Anatomie et Organogénèse	C.H.R.G.
FAURE Gilbert	Urologie	C.H.R.G.
FOURNET Jacques	Hépato-Gastro-Entérologie	C.H.R.G.
FRANCO Alain	Médecine Interne	C.H.R.G.
GIRARDET Pierre	Anesthésiologie	C.H.R.G.
GUIDICELLI Henri	Chirurgie Générale et Vasculaire	C.H.R.G.
GUIGNIER Michel	Thérapeutique et Réanimation Médicale	C.H.R.G.
HADJIAN Arthur	Biochimie	Faculté La Merci
HALIMI Serge	Endocrinologie et Maladies Métaboliques	C.H.R.G.
HOSTEIN Jean	Hépato-Gastro-Entérologie	C.H.R.G.
HUGONOT Robert	Médecine Interne	C.H.R.G.
JALBERT Pierre	Histologie-Cytogénétique	C.H.R.G.
JUNIEN-LAVILLAUROY Claude	O.R.L.	C.H.R.G.
KOLODIE Lucien	Hématologie Biologique	C.H.R.G.
LETOUBLON Christian	Chirurgie Générale	C.H.R.G.
MACHECOURT Jacques	Cardiologie et Maladies Vasculaires	C.H.R.G.
MAGNIN Robert	Hygiène	C.H.R.G.
MASSOT Christian	Médecine Interne	C.H.R.G.

MOUILLON Michel	Ophtalmologie	C.H.R.G.
PELLAT Jacques	Neurologie	C.H.R.G.
PHELIP Xavier	Rhumatologie	C.H.R.G.
RACINET Claude	Gynécologie-Obstétrique	Hopital Sud
RAMBAUD Pierre	Pédiatrie	C.H.R.G.
RAPHAEL Bernard	Stomatologie	C.H.R.G.
SCHAERER René	Cancérologie	C.H.R.G.
SEIGNEURIN Jean-Marie	Bactériologie-Virologie	Faculté La Merci
SELE Bernard	Cytogénétique	Faculté La Merci
SOTTO Jean-Jacques	Hématologie	C.H.R.G.
STOEBNER Pierre	Anatomie Pathologique	C.H.R.G.
VROUSOS Constantin	Radiothérapie	C.H.R.G.

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

46 avenue Felix Viallet
38031 GRENOBLE cedex

Tél. : 76.57.45.00

Année universitaire 1989

Président de l'Institut :
Monsieur Georges LESPINARD

Professeurs des Universités

BARIBAUD Michel	ENSERG	JAUSSAUD Pierre	ENSIEG
BARRAUD Alain	ENSIEG	JOST Rémy	ENSPG
BAUDELET Bernard	ENSPG	JOUBERT Jean-Claude	ENSPG
BEAUFILS Jean-Pierre	INPG	JOURDAIN Geneviève	ENSIEG
BLIMAN Samuel	ENSERG	LACOUME Jean-Louis	ENSIEG
BOIS Philippe	ENSHMG	LADET Pierre	ENSIEG
BONNETAIN Lucien	ENSEEG	LESIEUR Marcel	ENSHMG
BONNET Guy	ENSPG	LESPINARD Georges	ENSHMG
BRISSONNEAU Pierre	ENSIEG	LONGEQUEUE Jean-Pierre	ENSPG
BRUNET Yves	IUFA	LORET Benjamin	ENSHMG
CAILLERIE Denis	ENSHMG	LOUCHET François	ENSEEG
CAVAIGNAC Jean-François	ENSPG	LUCAZEAU Guy	ENSEEG
CHARTIER Germain	ENSPG	MASSE Philippe	ENSIEG
CHENEVIER Pierre	ENSERG	MASSELOT Christian	ENSIEG
CHERADAME Hervé	UFR PGP	MAZARE Guy	ENSIMAG
CHERUY Arlette	ENSIEG	MOHR Roger	ENSIMAG
CHOVET Alain	ENSERG	MOREAU René	ENSHMG
COHEN Joseph	ENSERG	MORET Roger	ENSIEG
COLINET Catherine	ENSEEG	MOSSIERE Jacques	ENSIMAG
CORNUT Bruno	ENSIEG	OBLED Charles	ENSHMG
COULOMB Jean-Louis	ENSIEG	OZIL Patrick	ENSEEG
COUMES André	ENSERG	PA ULEAU Yves	ENSEEG
CROWLEY James	ENSIMAG	PERRET Robert	ENSIEG
DARVE Félix	ENSHMG	PIAU Jean-Michel	ENSHMG
DELLA-DORA Jean	ENSIMAG	PIC Etienne	ENSERG
DEPEY Maurice	ENSERG	PLATEAU Brigitte	ENSIMAG
DEPORTES Jacques	ENSPG	POUPOT Christian	ENSERG
DEROO Daniel	ENSEEG	RAMEAU Jean-Jacques	ENSEEG
DESRE Pierre	ENSEEG	REINISCH Raymond	ENSPG
DOLMAZON Jean-Marc	ENSERG	RENAUD Maurice	UFR PGP
DURAND Francis	ENSEEG	ROBERT André	UFR PGP
DURAND Jean-Louis	ENSPG	ROBERT François	ENSIMAG
FAUTRELLE Yves	ENSHMG	SABONNADIERE Jean-Claude	ENSIEG
FOGGIA Albert	ENSIEG	SAUCIER Gabrièle	ENSIMAG
FONLUPTJean	ENSIMAG	SCHLENKER Claire	ENSPG
FOULARD Claude	ENSIEG	SCHLENKER Michel	ENSPG
GANDINI Alessandro	UFR PGP	SERMET Pierre	ENSERG
GAUBERT Claude	ENSPG	SILVY Jacques	UFR PGP
GENTIL Pierre	ENSERG	SIRIEYS Pierre	ENSHMG
GENTIL Sylviane	ENSIEG	SOHM Jean-Claude	ENSEEG
GREVEN Hélène	IUFA	SOLER Jean-Louis	ENSIMAG
GUEGUEN Claude	ENSIEG	SOUQUET Jean-Louis	ENSEEG
GUERIN Bernard	ENSERG	TROMPETTE Philippe	ENSHMG
GUYOT Pierre	ENSEEG	VINCENT Henri	ENSPG
IVANES Marcel	ENSIEG	ZADWORNY François	ENSERG

Personnes ayant obtenu le diplôme d'HABILITATION A DIRIGER DES RECHERCHES

BECKER Monique
BINDER Zdenek
CHASSERY Jean-Marc
CHOLLET Jean-Pierre
COEY John
COLINET Catherine
COMMAULT Christian
CORNUEJOLS Gérard
COULOMB Jean- Louis
COURNIL M.
DALARD Francis
DANES Florin
DEROO Daniel
DIARD Jean-Paul
DION Jean-Michel
DUGARD Luc
DURAND Madeleine
DURAND Robert
GALERIE Alain
GAUTHIER Jean-Paul
GENTIL Sylviane
GHIBAUDO Gérard
HAMAR Sylvaine
HAMAR Roger
LACHENAL D.
LADET Pierre
LATOMBE Claudine
LE HUY H.
LE GORREC Bernard
MADAR Roland
MEUNIER G.
MULLER Jean
NGUYEN TRONG Bernadette
NIEZ J.J.
PASTUREL Alain
PLA Fernand
ROGNON J.P.
ROUGER Jean
TCHUENTE Maurice
VINCENT Henri
YAVARI A.R.

Chercheurs du C.N.R.S

DIRECTEURS DE RECHERCHE CLASSE 0

LANDEAU	Ioan
NAYROLLES	Bernard

Directeurs de recherche 1ère Classe

ANSARA Ibrahim
CARRE René
FRUCHART Robert
HOPFINGER Emile
JORRAND Philippe
KRAKOWIAK Sacha
LEPROVOST Christian
VACHAUD Georges
VERJUS Jean-Pierre

Directeurs de recherche 2ème Classe

ALEMANY Antoine
ALLIBERT Colette
ALLIBERT Michel
ARMAND Michel
AUDIER Marc
BERNARD Claude
BINDER Gilbert
BONNET Roland
BORNARD Guy
CAILLET Marcel
CALMET Jacques
CHATILLON Chritiant
CLERMONT Jean-Robert
COURTOIS Bernard
DAVID René
DION Jean-Michel
DRIOLE Jean
DURAND Robert
ESCUDIER Pierre
EUSTATHOPOULOS Nicolas
GARNIER Marcel
GUELIN Pierre
JOUD Jean-Charles
KAMARINOS Georges
KLEITZ Michel
KOFMAN Walter
LEJEUNE Gérard
MADAR Roland
MERMET Jean
MICHEL Jean-Marie
MEUNIER Jacques
PEUZIN Jean-Claude
PIAU Monique
RENOUARD Dominique
SENATEUR Jean-Pierre
SIFAKIS Joseph
SIMON Jean-Paul
SUERY Michel
TEODOSIU Christian
VAUCLIN Michel
VENNEREAU Pierre
WACK Bernard
YONNET Jean-Paul

Personnalités agrées à titre permanent à diriger des travaux de recherche
(décision du conseil scientifique)

E.N.S.E.E.G
HAMMOU Abdelkader
MARTIN-GARIN Régina
SARRAZIN Pierre
SIMON Jean-Paul

E.N.S.E.R.G
BOREL Joseph

E.N.S.I.E.G
DESCHIZEAUX Pierre
GLANGEAUD François
PERARD Jacques
REINISCH Raymond

E.N.S.H.M.G
ROWE Alain

E.N.S.I.M.A.G
COURTIN Jacques

C.E.N.G
CADET Jean
COEURE Philippe
DELHAYE Jean-Marc
DUPUY Michel
JOUVE Hubert
NICOLAU Yvan
NIFENECKER Hervé
PERROUD Paul
PEUZIN Jean-Claude
TAIEB Maurice
VINCENDON Marc

Laboratoires extérieurs :

C.N.E.T
DEVINE Rodericq
GERBER Roland
MERCKEL Gérard
PAULEAU Yves

Situation particulière

PROFESSEURS D'UNIVERSITE

DETACHEMENT

ENSIMAG	LATOMBE	J..Claude	Détachement	21/10/1989
ENSHMG	PIERRARD	J.Marie	Détachement	30/04/1989
ENSIMAG	VEILLON	Gérard	Détachement	30/09/1990
ENSIMAG	VERJUS	J.Pierre	Détachement	30/09/1989
ENSPG	BLOCH	Daniel	Recteur à c/	21/12/1988

SURNOMBRE

INPG	CHIAVERINA	Jean	30/09/1989
ENSHMG	BOUVARD	Maurice	30/09/1991
ENSEEG	PARIAUD	J.Charles	30/09/1991

AVANT- PROPOS

Je tiens à exprimer ma reconnaissance à :

Monsieur Max VERDONE, Directeur de l'ENSERG, pour l'honneur qu'il me fait en présidant le jury de cette thèse.

Monsieur Bernard COURTOIS, Directeur de recherche au CNRS et directeur du laboratoire TIM3, pour m'avoir acceuilli dans l'équipe d'Architecture des ordinateurs et pour avoir pris la responsabilité de cette thèse.

Monsieur Alain GREINER, Professeur à l'Université de PARIS VI, pour avoir accepté d'être rapporteur de cette thèse et membre du jury.

Monsieur Yves GRUERE, Ingénieur en chef du département SPS de CSEE transport, pour avoir accepté d'être membre du jury.

Monsieur Mihalis NICOLAIDIS, chargé de recherche au CNRS, pour l'interêt permanent qu'il a porté à mon travail, pour son aide et sa collaboration et sa bonne humeur qui ont fait que mes travaux ont évolué dans les meilleures conditions.

Monsieur Paulo TEIXEIRA, Professeur à l'IST de l'université de LISBONNE, pour avoir accepté, malgré son éloignement d'être rapporteur de cette thèse et membre du jury.

Je ne saurais oublier dans mes remerciements tous les membres et ex-membres de l'équipe de recherche en architecture des ordinateurs et notamment Monsieur Alain GUYOT , Vladimir CASTRO ALVES et Kholdoun TORKI pour leur collaboration en des moments divers et à différents titres.

A mon père

RESUME

Actuellement, toutes les applications critiques mettant en jeu la vie humaine ne peuvent pas être assurées par des systèmes complexes utilisant des circuits intégrés répliqués; il est necessaire d'utiliser des composants discrets de sécurité intrinsèque, d'un encombrement et d'un coût prohibitifs.

Pour relever ce défi, le micro-contrôleur MAPS qui doit gérer la signalisation ferroviaire, bénéficie de l'intégration d'un circuit logique autotestable, en-ligne (duplication duale + parité) et hors-ligne, suivant le principe de la technique UBIST. Le MAPS dispose aussi d'une interface de sortie apte à produire des signaux de commande en fréquence, soit sûrs soit corrects. Il dispose également d'une interface d'entrée capable de n'accepter des signaux externes qu'après les avoir rendus sûrs ou corrects. Ces deux interfaces intégrées pour la première fois, sur la même puce que le circuit autotestable, sont "strongly fail-safe". Seules les communications avec l'extérieur se font par échange de messages fortement codés sans qu'aucun matériel ne soit rajouté.

En conséquence, l'étude que l'on présente permet d'apporter une nouvelle démarche de conception des systèmes hautement critiques, tout en assurant un degré de sécurité nettement plus élevé (détection de toutes les pannes triples) que celui donné par les systèmes actuels, et ceci pour un volume et un coût plus faibles.

MOTS-CLEF

circuits auto-contrôlables - test en-ligne - test hors-ligne - UBIST - micro-contrôleur - signalisation ferroviaire - strongly fail-safe - hypothèse de panne - duplication - sorties fréquence

A FAISABILITY STUDY OF A HIGH-SAFETY MICROCONTROLLER

ABSTRACT

Nowadays, all critical systems where human life is at stake cannot be controlled by complex devices using mass produced integrated circuits. Because of the need to use failsafe discrete components, the controller becomes cumbersome and prohibitively costly.

To tackle this problem, the MAPS microcontroller which is designed to drive railway signalling, uses an on-line (duplication and parity check) and off-line self-test integrated logic circuit according to the UBIST technique.

An output port is fitted to MAPS to produce fail-safe frequency modulated drive signals. It also uses an interface unable to accept external signals until after they have been recast into a failsafe format.

These two interfaces have been integrated for the first time on the same chip as the self-checking circuit and are strongly fail-safe. Communications with the outside world are only made using highly coded message passing without extra hardware.

Consequently, we show in this study, a new approach to the design of highly critical systems, yielding at the same time a higher safety factor (detection of all triple faults) than that given by present systems and this for less bulk and lower cost.

KEYWORDS

self-checking circuits - on-line test - off-line test - UBIST - microcontroller - railway signalling - strongly fail-safe - fault hypothese - duplication - frequency outputs

SOMMAIRE

INTRODUCTION

Les progrès de la technologie au cours de ces dernières années ont entraîné un formidable essor des circuits intégrés à tel point que les techniques de test externe couramment utilisées sont devenues insuffisantes pour couvrir l'ensemble des pannes possibles susceptibles de se produire dans les circuits intégrés. Cette situation est donc tout à fait incompatible avec une application de sécurité quelconque puisque le mauvais fonctionnement d'un élément du circuit, aussi infime soit il peut engendrer des situations critiques ou même catastrophiques. Une défaillance, dont l'effet local n'est pas critique peut, si elle n'est pas détectée, contaminer d'autres unités du système, et de ce fait produire une catastrophe. Il est donc essentiel de considérer le test des circuits dès leur conception pour assurer une détection d'erreur immédiatement après leur apparition. Cette restriction est la condition sine qua non de toute utilisation de circuits intégrés dans des systèmes complexes dont la fonction est de maintenir la sécurité.

Position du problème

Cette étude a pour but d'introduire les circuits intégrés de sécurité pour traiter les automatismes ferroviaires. Il va sans dire, que cela n'est concevable que si on peut assurer l'entière sécurité des voyageurs quelles que soit les conditions d'exploitation.
La démarche que l'on suit nous oblige à faire évoluer le concept de sécurité intrinsèque des systèmes de sécurité actuels à base de relais vers un concept de sécurité probabiliste pour des circuits intégrés aux fonctions multiples et variées.
Cette évolution de la sécurité est nécessaire car les performances que l'on demande à un système de gestion ferroviaire se diversifient de plus en plus (Vitesse élevée, intervalle entre trains réduit, comptage d'essieux,...) et le bon fonctionnement d'un tel système n'est possible qu'avec une puissance de traitement importante que seules les techniques nouvelles (microprocesseur) peuvent assurer.

Les progrès réalisés par la technologie et la conception des composants semi-conducteurs (SSI -> MSI -> LSI -> VLSI) ont permis de disposer d'une puissance de calcul suffisante mais au détriment de la sécurité intrinsèque qui a été perdue en MSI. La seule possibilité d'exploiter des microprocesseurs dans des systèmes critiques est donc d'assurer que l'apparition d'une panne dangereuse ne survient qu'avec une probabilité telle qu'une catastrophe puisse être raisonnablement jugée improbable.

L'augmentation de la sécurité n'est concevable que si l'on bénéficie aussi d'une bonne fiabilité car même si la sécurité maximum préconise l'immobilité il faut cependant assurer le déplacement, dans les meilleures conditions (confort, temps ...) de l'usager. Cela revient à dire qu'il faut trouver le bon dosage entre puissance de traitement, sécurité et fiabilité.

Le MAPS (micro-contrôleur programmable de sécurité) a donc la lourde tâche de remplacer les systèmes de sécurité utilisés actuellement pour commander les modules à la voie et dialoguer avec le poste d'aiguillage.Pour cela, nous avons donc proposé d'implémenter conjointement un micro-contrôleur self-checking et des interfaces strongly fail-safe sur le même circuit. La solution proposée détecte les pannes triples et assure donc ainsi une très grande protection vis à vis des pannes de mode commun qui restent difficiles à modèliser. L'originalité de ce circuit réside principalement dans l'intégration d'interface strongly fail-safe , d'un double codage du micro-contrôleur et de l'utilisation de générateurs spécifiques pour le test hors-ligne.

L'étude qui a été menée fait suite aux motivations formulées par la CSEE (Compagnie de signaux et d'entreprise électrique) qui voulait savoir comment faire évoluer un produit commercial (gestionnaire de réseau ferroviaire) en fonction des nouvelles théories de test élaborées au sein du laboratoire TIM3. Ce travail est donc basé sur des données concrètes pour réaliser le système MAPS qui a les mêmes fonctionnalités mais dont la conception est entièrement repensée .

Ces données sont rappelées au chapitre I ainsi que les alternatives de conception que l'on pouvait envisager. Cette étude qui se veut un exemple de ce que l'on peut faire de plus performant pour atteindre une très haute sécurité a permis , rapidement, de ne retenir qu'une seule solution, la technique UBIST.

Les chapitres II et III donnent les bases théoriques de cette technique. Cependant, la technique UBIST n'a tout son intérêt que si elle est associée à la propriété strongly fail-safe au niveau des interfaces.

Le chapitre IV est réservé à la présentation des bases théoriques permettant d'atteindre cette propriété.

Avant d'appliquer les théories établies dans les 3 chapitres précédents, il est indispensable de savoir quels seront les organes vitaux de ce circuit. Cela est précisé dans le chapitre V.

Les modifications de l'architecture en fonction des tests et de la sécurité que l'on veut apporter au circuit sont détaillées au fil des 3 chapitres qui suivent. Le chapitre VI est plus particulièrement réservé au test en-ligne du circuit alors que le chapitre VII est réservé à la mise en place du test hors-ligne pour ainsi atteindre, grâce à la technique UBIST le TSC goal. Le chapitre VIII parachève cette description en détaillant les interfaces d'entrée et de sortie qui ne doivent délivrer que des informations correctes ou sûres, et ceci quel que soit l'état du circuit (Strongly fail-safe).

Le chapitre IX essaie, grâce à des données théoriques, de donner des chiffres qui traduisent la taille et la qualité de la sécurité que l'on peut espérer. Ces chiffres ne sont, bien entendu, que des ordres de grandeur.

CHAPITRE I

DESCRIPTION D'UN RESEAU FERROVIAIRE

I.1 INTRODUCTION

Les spécifications que nous allons exposer dans ce chapitre donnent une description des contraintes externes préalables au développement d'un micro-contrôleur programmable de sécurité. Ces spécifications décrivent principalement les modules à la voie d'un poste d'aiguillage informatisé(PAI).

Cette description nous permettra ensuite de donner des alternatives de conception d'un système de sécurité.

I.2 GENERALITES

Les fonctions prises en compte par un poste d'aiguillage peuvent être classées selon leur niveau de sécurité.

I.2.1 : Fonctions de commande locale ou décentralisées

I.2.1.1: Fonctions d'aide à l'exploitation (niveau 0.1)

Le niveau réalise :

La programmation et la commande automatique des itinéraires

Le suivi des trains

Le télé affichage sur tableau de contrôle

I.2.1.2: Fonctions de commande et de contrôle de la signalisation (niveau 0.2)

Ce niveau réalise :

- La prise en compte des commandes de l'exploitant
- La présentation des contrôles de l'exploitant
- La télé transmission des commandes et des contrôles

I.2.2 : Fonctions d'enclenchement (niveau 1)

Ce niveau réalise :

- L'enclenchement et le contrôle des itinéraires
- L'espacement des trains en gare et en pleine ligne
- La protection du personnel
- La protection des caténaires privés de courant

- L'annonce et le réarmement des passages à niveau
- L'annonce des circulations aux chantiers

I.2.3 : Fonctions gestion des équipements en campagne (niveau 2)

Ce niveau correspond à l'application " Module à la voie" qui fera plus spécialement l'objet de cette thèse. Les principales fonctions à réaliser sont les suivantes.

- Détection et présence des trains par circuit de voie
- Détection de passage à niveau par détecteur ponctuel (pédale)
- Commande et contrôle des aiguilles
- Présentation des informations aux mécaniciens par :
- Commande et contrôle des panneaux lumineux
- Transmission voie-machine (TVM)
- Crocodile
- Détonateur.
- Commande des feux, de la sonnerie et des barrières des passages à niveau.
- Autorisations de manoeuvre, d'annulation de transit.

Ces fonctions nécessitent l'intégration de :

- Temporisations et minuteries
- Commutation jour/nuit
- Commandes perdues
- Commandes décalées ou en cascade
- Clignotement
- Filtrages et mémorisations.

I.2.4 : Equipements en campagne (niveau 3)

Ce niveau correspond aux capteurs et aux organes de commande ou de contrôle à la voie. Il est nécessaire d'y ajouter la partie signalisation en cabine, liée à la TVM. C'est ce niveau qui intervient directement pour donner la direction, régler la vitesse et connaître la position des trains. Nous y trouvons les aiguilles, les signaux ainsi que les détecteurs, les contrôleurs d'allumage de lampes, etc ...

I.2.5 : Fonctions de sécurité

Les fonctions de niveau 0.1 ne sont pas de sécurité

Les fonctions de niveau 0.2 doivent être sûres

Les fonctions de 1,2,3 sont de sécurité

I.3 SIGNALISATION CONVENTIONNELLE

Chaque module à la voie est défini par des équipements en campagne qu'il commande ou contrôle. Une logique interne personnalise chaque module.

I.3.1: Module panneau

Il assure la gestion de la logique d'allumage des lampes de panneau. Il est composé d'un panneau principal et d'un tableau annexe, si cela est nécessaire.

I.3.1.1: Panneaux principaux

Il n'existe aucun standard défini, mais il est néanmoins possible de définir une configuration minimale et maximale. La configuration maximale comporte 8 commandes de lampes dont 4 sont contrôlées (voir tableau n°1). La configuration minimale comporte 2 commandes de lampes.

signal		tension	puissance	contrôle
C	carré	7,2V	1x15W	non
S	sémaphore	7,2V	1x15W	non
M	indication d'erreur	7,2V	1x15W	non
A	avertissement	7,2V	1x15W	oui
VL	voie libre	7,2V	1x15W	oui
R	ralentissement	7,2V	2x15W	oui
RR	rappel ralentissement	7,2V	2x15W	oui
oeilleton	panneau actif	8V	1x3W	non

Tableau n° 1 : Recensement des signaux constituant un panneau

Cependant une configuration normalisée est souvent utilisée en pleine ligne, c'est un bloc automatique lumineux à 3 ou 4 signaux. Ces signaux sont les signaux S, A, VL et éventuellement l'oeilleton du tableau n° 1.

La logique interne d'un panneau est définie de la manière suivante :

- La temporisation du contrôle par rapport à la commande
- Le clignotement éventuel d'un ou plusieurs signaux
- Le report de signalisation en cas de décontrôle

Si le contrôle d'un signal n'a pas été fait on doit passer à un mode de fonctionnement plus restrictif, cette dégradation peut aller jusqu'à l'arrêt complet.

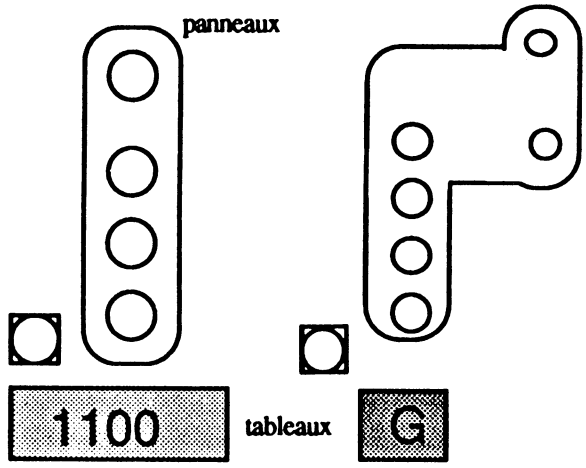


Schéma n° 1 : 2 types de panneaux avec tableaux annexes

I.3.1.2 : Tableaux annexes

Ce sont des indications qui sont implantées sur le panneau principal. Leur grande diversité et leur nombre de commandes ne nous permettent pas de donner une configuration type mais simplement un exemple(voir schéma n° 1). Si l'association panneau plus tableau demande un très grand nombre de signaux, il suffit d'implanter deux modules à la voie. De même que pour les panneaux, certains signaux demandent des commandes de sécurité et des contrôles tandis que d'autres ne nécessitent aucun contrôle. Tout décontrôle d'un signal du tableau annexe commande le report d'un signal du tableau principal.

I.3.1.3 : Crocodile, détonateur et balise commutable

Ces dispositifs sont utilisés pour envoyer une ou plusieurs informations au mécanicien lors de la détection du train. L'information sonore délivrée par les crocodiles ou les détonateurs est déclenchée si le panneau concerné demande l'arrêt. Dans le cas de la balise commutable, l'information (1 parmi 5) est transmise en cabine et permet le contrôle de la vitesse du train conformément à l'état du panneau concerné.
La logique interne associée à ces dispositifs définit l'envoi des commandes en fonction de la détection de passage du train et l'état des signaux.

I.3.2 : Module passage à niveau

Il assure la commande des feux, des sonneries et des mécanismes des barrières de passage à niveau. Le mécanisme de chaque barrière est contenu dans un coffret. Ce mécanisme comprend le moteur, le relais de commande du moteur, le relais de rupteur, la résistance de freinage, la résistance de chauffage, ainsi qu'un ensemble de 8 bagues calibrées qui délivrent les informations suivantes : montée, descente, freinage, contrôle fermeture, commande d'ouverture.

De manière générale les modules à la voie commandent et contrôlent 4 barrières, 2 sonneries et les 4 feux correspondants. Il faut donc commander lors de l'annonce :

- 4 feux en clignotant (19,4V-25W)
- 2 sonneries (24V-4W)
- 2 abaissements de barrière avec temporisation (24V-5W)
puis 2 levers de barrière

Les contrôles à effectuer sont les suivants :

- l'annonce par les pédales sur chaque voie.
- le réarmement éventuel des pédales
- les circuits de voies
- la position des barrières

Il est cependant possible de reprendre localement la commande du passage à niveau à partir de 4 modes de fonctionnement et avec l'aide de voyants. La reprise en local est bien entendu implantée à l'extérieur du module.

La logique interne définit l'envoi des commandes en cascade, le clignotement des feux et le réarmement de l'annonce (pédale).

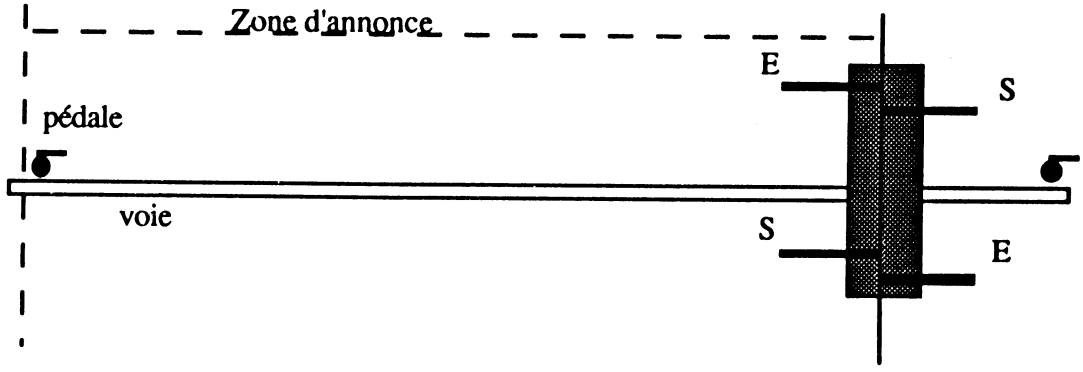


Schéma n° 2 : description d'un passage à niveau

I.3.3 : Le module aiguille

Il assure la commande du moteur de l'aiguille, le contrôle de la position et le verrouillage éventuel des lames d'aiguille. Toutes les aiguilles comportent :

- 2 commandes interverrouillées
- 2 contrôles de position

La puissance demandée par les moteurs continus ou alternatifs (de 110 à 380V, de 1,7 à 90 A) nécessite l'utilisation de relais de découplage intermédiaires.

La logique interne définit :

- l'interverrouillage des 2 commandes entre elles, avec des contrôles de positions respectifs et, éventuellement, la détection du train.
- la durée des commandes perdues
- la temporisation associée

I.3.4 : Le traitement des pédales

La pédale est, pour le module à la voie, une entrée spécifique (différente du tout ou rien des autres entrées) qui permet la détection ponctuelle d'un train. Elle peut se trouver associée à un passage à niveau, à un panneau, seule ou proche d'une aiguille. Son principe est basé sur la détection de l'apparition et de la disparition de deux fréquences précises (50Hz et 39Hz) au passage de chaque essieu du train.

Les blocs de traitement utilisés actuellement peuvent détecter les trains dans un ou deux sens et l'état du relais de sortie peut être négatif (désactivation) ou positif (activation).
Le module à la voie doit traiter tous les cas, c'est à dire que la détection du sens doit être bi-orientée, par contre comme le relais de sortie est remplacé par une mémoire interne, il n'y aura plus à distinguer le négatif du positif.

La logique interne acquiert les 4 temps de la séquence (disparition du 39Hz, disparition du 50Hz, réapparition du 39Hz et réapparition du 50Hz) dans les 2 orientations possibles. Elle effectue les temporisations de filtrage et d'inhibition, ainsi que la remise à zéro et le comptage/décomptage d'essieux.

I.3.5 : Divers

Le module à la voie commande aussi 1 à 4 mécanismes de caténaire et contrôle leurs positions. Chaque mécanisme comprend un moteur, les relais de commande, une résistance de chauffage et 8 contacts de positions. L'alimentation des caténaires est commandée par le poste d'aiguillage.

Le module à la voie peut commander des résistances de chauffage au niveau de chaque module. Cette fonction n'est pas de sécurité.

Le module" protection et annonce aux chantiers" permet d'assurer une protection d'un chantier par action sur la signalisation.

I.3.6 : Configuration d'une installation

Un poste d'enclenchement peut gérer jusqu'à 64 modules à la voie mais dans une gare importante plusieurs postes d'enclenchement dialoguent ensemble pour gérer jusqu'à 400 modules à la voie.

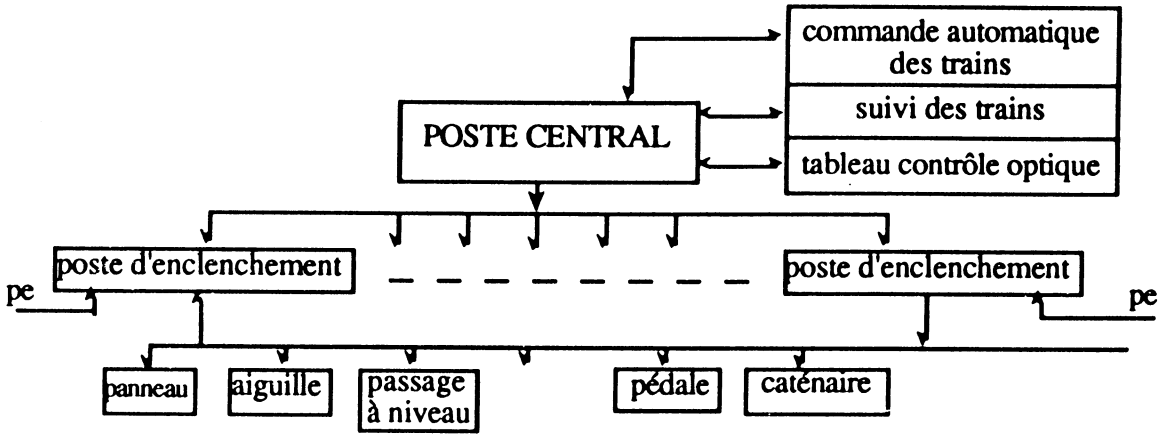


Schéma n°3 : Configuration d'une installation importante

I.3.7 : Aide à la conduite

Emission à la voie de la TVM "ponctuelle"

Le module émission à la voie participe à l'élaboration et à l'émission d'information à l'intention de la cabine du train. Il comprend une sortie série spécifique (avec relecture). Ces informations transitent par la voie et sont traitées par l'équipement en cabine pour le contrôle de vitesse du train. Son principe de base consiste à ce que l'équipement au sol fournisse à chaque train :

- Les indications variables des signaux d'annonce et d'exécution,
- Les paramètres invariants de description de la zone de voie contrôlée

pour lui permettre d'élaborer, à partir de ses propres paramètres et en fonction de sa localisation courante, sa courbe de contrôle.

Le module à la voie devrait effectuer :

- L'acquisition des indications variantes
- L'élaboration des données numériques complètes
- La comparaison de ces données
- La réception des données retransmises par l'équipement en cabine
- La comparaison des données émises et des données reçues.

Les indications variantes sont acquises sur les panneaux de signalisation. Les données numériques sont élaborées en rajoutant aux indications invariantes mémorisées dans le module à la voie un code de redondance cyclique.

I.4 SPECIFICATION D'UTILISATION

L'inventaire qui vient d'être fait au chapitre précédent permet, après unification, de définir la nature et le nombre d'interfaces nécessaires pour couvrir les différentes applications.
La solution la plus intéressante est de disposer d'un mini-automate possédant :
- 8 entrées et 8 sorties
Ce maximum permet de gérer un panneau complexe ou d'acquérir les variants de la TVM, néanmoins, dans certains cas, 4 entrées et 4 sorties sont suffisantes.

- Interface d'entrées/sorties

Entrées lentes:

Les différentes entrées "tout ou rien" répertoriées sont du type :

- Contact de relais libre de potentiel
- Courant/tension pour contrôle de filament
- Fréquence pour les pédales

La dynamisation de ces entrées par rebouclage d'une séquence pseudo-aléatoire de 32 bits sera détaillée par la suite.

- Les entrées rapides

Elles auront pour fonction de compter le nombre d'essieux et d'indiquer le sens des trains. Elles devront donc être scrutées à une fréquence plus rapide que les entrées classiques. Cette fonction sera entièrement réalisée, de manière interne par le circuit, elle sera détaillée par la suite.

- Les sorties faible, moyenne ou forte puissance

Les sorties commandent des relais de différentes puissances à l'aide de fréquences toutes différentes les unes des autres.

- Une interface de communication paramètrable capable de s'adapter à différents réseaux ou bus.
La sécurité de l'information est dans ce cas supportée par le codage. Pour cette étude nous choisirons le message le plus contraignant à savoir 9 octets de codage pour un octet d'information.

-Une fonction du type automate programmable très simplifiée.

I.5 ALTERNATIVES DE CONCEPTION

Outre la résolution de toutes les opérations demandées par les spécifications, le circuit que l'on veut réaliser doit garantir une sécurité au moins aussi importante que les systèmes à base de relais actuellement utilisés. Ce genre de circuit demande, de manière générale, l'intégration de deux chaînes, l'une de commande et l'autre de surveillance. La chaîne de surveillance n'a pour fonction

que de vérifier constamment le bon fonctionnement de la chaîne de commande. Tout ce que contient la chaîne de surveillance augmente donc le circuit, soit au niveau matériel, soit au niveau du temps d'exécution.

Etant donné que les fonctions à traiter par les automatismes ferroviaires deviennent de plus en plus importantes et complexes, il paraît évident que l'utilisation de circuit du type microprocesseur va se généraliser. A ce niveau, plusieurs filières s'offrent à nous :

I.5.1 : La filière boîte noire

C'est la filière la plus couramment par les industriels et l'interface est réalisée à base de relais de sécurité, ce qui la rend très volumineuse. Dans cette filière, le microprocesseur est considéré comme une boîte noire, c'est à dire que l'on ne cherche pas à connaître sa structure interne et la sécurité est assurée par une redondance de matériel. Cette technique permet d'utiliser les microprocesseurs du commerce qui ont été conçus pour des applications très variées et non sécuritaires. Pour assurer la sécurité, plusieurs solutions peuvent être envisagées :

- L'utilisation de deux microprocesseurs et d'un comparateur.

Cette solution a été adoptée par Daimler Benz [KLO 87] pour résoudre un problème de signalisation ferroviaire. Les sorties des deux microprocesseurs ne sont validées que si elles donnent le même résultat, sinon le système s'arrête en laissant ses sorties dans un état sûr. Ce système peut, soit ne contrôler que les sorties, soit contrôler aussi les variables internes (ceci n'est pas le cas pour les micro-processeurs du commerce) et permettre ainsi une détection plus précoce, donc plus sûre des erreurs .

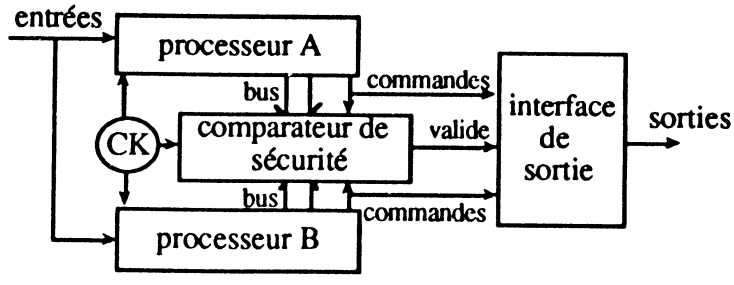


Schéma n° 4 : Filière boîte noire biprocesseur

- L'utilisation de plusieurs systèmes et un circuit de vote majoritaire.

Cette solution a été retenue par le consortium British railway, CEG et Westinghouse pour la partie centralisée et les modules décentralisés sont dupliqués. Si l'on utilise trois processeurs, les sorties ne sont validées que si deux des trois processeurs donnent des résultats identiques. Cette structure a l'avantage d'être reconfigurable en un système à deux processeurs comme décrit précédemment puis à nouveau reconfigurable en un système 2 parmi 3 après l'installation d'une

nouvelle carte. Il est bien entendu essentiel de ne pas perdre le contexte acquis entre les deux reconfigurations. Cette structure est donc beaucoup plus fiable.

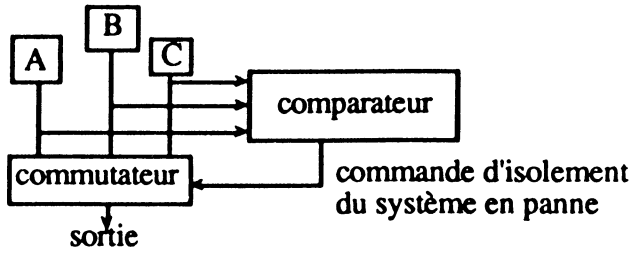


Schéma n° 5 : Filière boîte noire à 3 processeurs

L'inconvénient des structures à plusieurs processeurs est qu'elles sont incapables de détecter des pannes (mode commun) survenant simultanément dans 2 processeurs.

- L'utilisation d'une redondance logicielle

On peut utiliser deux logiciels différents dans deux systèmes différents ou bien deux logiciels qui fonctionnent alternativement dans un système unique (redondance temporelle) puis on compare les deux résultats.

On peut aussi utiliser la redondance par codage mathématique, principalement dans les transmissions de messages entre l'équipement au sol et l'équipement embarqué.

Les problèmes rencontrés sont souvent dûs aux codages des logiciels, diverses méthodes sont alors appliquées :

- Programme simple en langage assembleur
- Utilisation de programmes modulaires en évitant les boucles
- Eviter les interruptions en les remplaçant par des scrutations cycliques
- Choisir la structure du logiciel en fonction du matériel choisi (comparaison du déroulement alternatif de deux logiciels identiques ou utilisation de deux logiciels conçus indépendamment l'un de l'autre).

La difficulté provient principalement du fait que l'utilisateur ne connait pas la structure du microprocesseur et qu'il est donc incapable de prévoir toutes les pannes possibles.

Le codage mathématique (codes arithmétiques) est quant à lui indépendant du type de microprocesseur. Ce codage permet de calculer le pourcentage d'erreurs indétectables. Le niveau de sécurité peut être amélioré en augmentant le nombre de bits de codage, mais le temps de codage augmente rapidement avec le niveau de sécurité requis. Cela peut être un facteur limitatif au cas où l'aspect temps réel deviendrait critique. D'autre part, à cause de son principe probabiliste, ce codage

ne permet pas la détection à 100% des pannes simples, doubles etc... Les erreurs dépendent du type de circuit qui est utilisé pour le traitement des données, par conséquent, le pourcentage réel des erreurs détectables dépendra du matériel et le danger provient du fait que dans certains cas la probabilité réelle de détection pourrait être nettement inférieure à la probabilité calculée.

I.5.2 : La filière boîte grise

Cette technique consiste à utiliser un microprocesseur spécifique dans lequel on se donne des moyens de commandabilité et d'observabilité. Les tests ont généralement lieu pendant les intervalles où le système n'a aucune opération à effectuer grâce à l'injection d'entrées convenables. Le procédé généralement le plus utilisé est le compactage des informations.

En conclusion, suite à des constats qui ont pu être faits par divers utilisateurs de systèmes de transport, on peut remarquer que

- La réalisation d'un test exhaustif n'est quasiment jamais possible lorsque l'on utilise la redondance matérielle.
- Les temps de test ou de codage ont tendance à devenir prohibitifs face aux nouvelles exigences des systèmes récents.
-La sécurité n'est jamais transparente à l'utilisateur.

I.5.3 : La filière boîte blanche

Compte tenu des restrictions des systèmes présentés ci-dessus et de la sécurité que l'on doit apporter au circuit, nous nous sommes plus particulièrement intéressés à cette filière pour la réalisation du projet MAPS.
L'idée générale consiste à utiliser un microprocesseur spécifique en prenant en compte les pannes dès la conception. Pour cela on utilise des technique du type DFT (design for testability). C'est à dire que la chaîne de surveillance est directement intégrée avec le circuit. Cette filière traite aussi bien du test en-ligne que du test hors-ligne.

L'autotest intégré

Les performances de ces techniques peuvent être améliorées en intégrant dans le circuit des mécanismes nécessaires à son propre test, c'est la technique BIST (Built-In Self Test). C'est à dire que l'on incorpore dans le circuit la génération de vecteurs de test et l'observation des sorties. Les différents générateurs de vecteurs de test et les analyseurs de signature (observation des réponses par compactage) sont réalisés avec des registres à décalage à rebouclage linéaire (linear feedback shift register = LFSR).

Cette technique simplifie énormément le test des circuits complexes car le temps d'application est court, les circuits sont contrôlés à vitesse d'utilisation, l'initialisation est rapide et la phase d'élaboration des vecteurs de test est évitée (génération de vecteurs exhaustive ou pseudo-aléatoire). par contre, l'augmentation de surface peut être importante (voir schéma n°6).

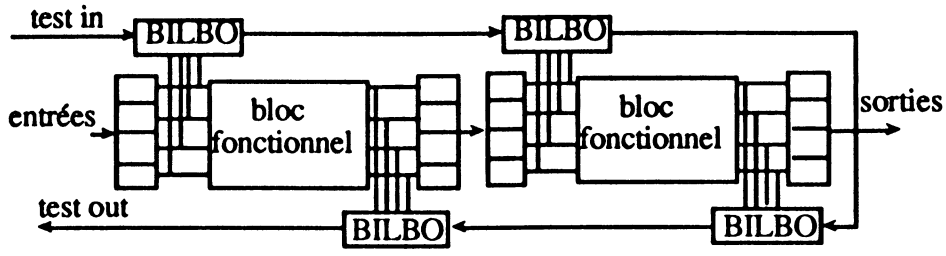


Schéma n° 6 : Autotest intégré

Généralement, les générateurs et les analyseurs sont combinés en blocs appelés BILBO (built in logic bloc observer) proposés par [KON 79].

Il faut cependant remarquer que ces techniques ne permettent que d'effectuer un test hors-ligne ou un test en-ligne discontinu. Ces solutions sont donc insuffisantes pour des systèmes de sécurité dans lesquels il s'agit de détecter des erreurs très rapidement, avant qu'une contamination du circuit provoque une situation catastrophique en sortie.
Le domaine d'application du circuit MAPS nécessite une très haute sécurité, il est donc impératif de pouvoir détecter l'occurrence d'une panne immédiatement au cours du fonctionnement du circuit; c'est le test en-ligne intégré, assuré par la technique dite d'auto-controlabilité ou "self-checking".
Cependant, en fonctionnement normal, certaines parties du circuit peuvent n'être que très rarement sollicitées, cela favorise la latence des pannes. La solution est donc d'intégrer conjointement le test en-ligne et le test hors-ligne pour aboutir à la technique UBIST (Unifited Built-In Self Test).
La description de cette technique sera détaillée dans le chapitre suivant mais ont peut déjà dire que dans cette filière la sécurité est transparente à l'utilisateur puisque sa démonstration est faite dès la conception, par contre la non tolérance aux fautes est maintenue.
Le second avantage du circuit que l'on propose est que les interfaces fail-safe sont aussi totalement intégrées et ceci pour la première fois.

I.6 CONCLUSION

Ce chapitre avait pour but de présenter l'environnement dans lequel le circuit MAPS devrait s'insérer. Il a aussi permis d'exposer les diverses alternatives qui permettent de concevoir des

systèmes de sécurité. Toutefois, on s'est rapidement aperçu que toutes les techniques qui existent actuellement ne satisfont pas les très sévères critères de sécurité imposés aux transports ferroviaires.

En conclusion on peut affirmer que seul l'autotest intégré est capable de donner satisfaction, reste à prouver que la probabilité d'apparition d'évènement catastrophique est inférieure à celle des systèmes à relais actuellement utilisés.

CHAPITRE II

TECHNIQUES DE TEST

II.1 INTRODUCTION

L'évolution rapide de la micro-électronique, liée aux progrès réalisés dans le domaine de la technologie empêche désormais les techniques de prévention couramment utilisées jusqu'à aujourd'hui (test externe) de couvrir l'ensemble des pannes possibles dans les circuits intégrés logiques.

Cela est d'autant plus critique lorsque l'application est dite de sécurité car la défaillance d'un système peut être à l'origine de situations critiques ou catastrophiques pouvant entraîner la mort.

Dans ce cas, il est nécessaire de signaler la présence de la première sortie erronée. Cette mission vitale n'est en aucune façon remplie par les techniques classiques.

Pour satisfaire un tel besoin, il faut prendre en considération le test en-ligne du circuit dès la conception. Cette technique a l'avantage de couvrir, avec un apport de matériel, les modes de défaillances réels [FER 88]. Elle est basée sur des techniques de codage et des propriétés mathématiques des différents blocs du circuit et a pour but de détecter la première manifestation d'une erreur (totally self-checking goal).

Toutefois, le test en-ligne du circuit ne peut être effectif que si certaines conditions, très restrictives sont satisfaites.

Ce chapitre va donc s'attacher à présenter théoriquement sous forme de rappels et de définitions les techniques de test que l'on va appliquer tout au long de la conception du circuit MAPS.

II.2 LES HYPOTHESES DE PANNES POUR LE TEST EN-LIGNE

II.2.1 : Historique

Pendant longtemps, le modèle pris en compte pour le test des circuits intégrés a été celui du collage logique [FRI 71]. Dans ce modèle, on représente les circuits par des portes logiques et on considère qu'une entrée ou une sortie d'une porte prend constamment la valeur logique 0 ou 1. Ce modèle n'est pas représentatif de toutes les pannes réelles des circuits intégrés [GAL 80], [WAD 78]. Par conséquent, de nouveaux modèles sont apparus dans lesquels les circuits sont représentés au niveau électrique par des réseaux de transistors MOS et on considère comme pannes possibles les collages des lignes à une valeur logique, des collages de MOS passant ou ouvert et des courts-circuits [ELZ 81], [MAL 82], [CHI 82].

En ce qui concerne le projet MAPS, on considère l'implémentation réelle des circuits, telle que décrite par le dessin des masques, dans laquelle on considère des lignes de diffusion, de polysilicium et d'aluminium, des croisements entre une ligne de polysilicium et une ligne de diffusion pour former un MOS, des contacts entre deux lignes de niveaux différents etc.... On considère donc les défaillances de ces éléments, c'est à dire, la coupure d'une ligne, un court-circuit entre deux lignes, un contact défaillant, un MOS défaillant, etc... A ce niveau on dispose de la classification de pannes établie dans [COU 81] qui est basée sur l'analyse des mécanismes de défaillance liés à la durée de vie des circuits, ce qui correspond tout à fait aux besoins du test en-ligne.
Comme il est démontré dans [COU 81], les mécanismes de défaillance liés à la durée de vie des circuits ne peuvent pas produire des courts-circuits entre deux lignes de polysilicium ou entre deux lignes appartenant à des niveaux différents (diffusion -métal). D'autre part, les mécanismes de défaillance liés à la durée de vie des circuits sont très lents. Il est donc réaliste de considérer que pour le test en-ligne, les pannes matérielles sont simples (une seule panne présente). Toute la réussite de la méthode dépend donc de l'utilisation d'un modèle de pannes simples qui représente bien les défaillances réelles. Seule la classe I issue de la classification donnée dans le tableau n°1 est donc à considérer.

II.2.2 : Hypothèses de panne

classe 0	classe I	classe II
un défaut simple un contact coupé un MOS s-on un MOS s-open un alu coupé un poly coupé une diff coupée une grille flottante	classe 0 + courts-circuits entre 2 alu les plus proches géographiquement de même pour la diffusion	classe I + courts-circuits entre 2 alu quelconques de même pour la diffusion défauts multiples

Tableau n°1 : classe des hypothèses de pannes

S-ON ou collé passant : Ce sont des transistors passants en permanence, quelle que soit la tension de grille. Ce type de panne repose sur deux origines, la modification des caractéristiques technologiques et l'influence des effets de canaux courts (perçage entre drain et source, injection de charges dans l'oxyde de grille, modification de la densité de charge dans la zone du canal, réduction de la longueur du canal, toute panne modifiant la tension de seuil ou/et la conductivité)

S-OPEN ou collé ouvert : Un transistor S-open est soumis à une panne logique qui empêche un réseau de conduire lorsque l'autre est dans un état de non conductivité. La sortie d'une porte CMOS se trouve alors en état de haute impédance et mémorise la valeur précédente (un circuit combinatoire devient séquentiel). L'origine de ce type de défaut peut être localisée, soit au niveau transistor (coupure de canal, de contact drain ou source, de grille avec présence d'une charge active à l'interface active du transistor) soit, au niveau des connexions entre transistors, commandes et alimentation.

Les hypothèses de pannes données précédemment pour la technologie NMOS sont aussi valables pour la technologie CMOS étant donné que les mécanismes de défaillance sont les mêmes dans les deux cas. Par contre, les conséquences de ces hypothèses peuvent être différentes.
Ces hypothèses de pannes ne prennent pas en compte les tests de mise au point et de fin de fabrication.

La classe I précédemment citée peut se diviser en 5 grandes catégories.

A)
- UN CONTACT DEFAILLANT
- UN MOS S-OPEN
- UN MOS S-ON
- LA COUPURE D'UNE LIGNE (sauf alimentation)
- UN COURT-CIRCUIT ENTRE UNE LIGNE ET L'ALIMENTATION
- UN COURT-CIRCUIT ENTRE DEUX LIGNES QUI CONVERGENT EN UNE SEULE

La catégorie A ne regroupe que des pannes qui produiront une erreur simple dans une ligne interne du circuit . Les s-open feront l'objet d'un cas particulier puisqu'ils transforment un circuit CMOS combinatoire en un circuit séquentiel.

B)
- LES COURTS-CIRCUITS ENTRE DEUX LIGNES QUI NE SONT PAS DES ALIMENTATIONS ET QUI NE CONVERGENT PAS EN UNE MEME LIGNE

* Si les deux lignes ont la même valeur, il n'y aura pas d'erreur.
* Si les deux lignes ont des valeurs différentes, les deux lignes auront une tension comprise entre Vdd et Vss suivant la résistance des réseaux N et P qui connectent les deux lignes à Vss ou Vdd.

- Si l'erreur est logique ce sera une erreur simple
- Si l'erreur est non logique, deux cas peuvent se produire
- soit il n'y a pas d'erreur (l'erreur est redressée)
- soit l'erreur est double (spécifique au CMOS statique) suivant les tensions de seuil des portes placées immédiatement après.

On verra par la suite que dans l'implémentation du circuit MAPS, les PLAs les ROMs et les RAMs sont implémentés en CMOS dynamique, ainsi le problème des erreurs doubles dues aux courts-circuits ne se posera pas.Une règle particulière se reportant à ce cas est présentée dans [NIC 87].
Cette solution à aussi l'av. · d'être plus facilement testable [JHA 88], [JHA 89] et [NIC 90] , cette affirmation est détaillée dans l'annexe III.

C)
- COUPURE D'UNE LIGNE D'ALIMENTATION (Vdd ou Vss)
L'erreur devient une erreur unidirectionnelle aux sorties des portes alimentées par cette ligne.

D)
- UN COURT-CIRCUIT ENTRE DEUX LIGNES Vss OU DEUX LIGNES Vdd
Ce cas ne crée pas d'erreur (sauf dans des situations particulières qui sont détaillées dans la suite de ce document).

E)
- UN COURT-CIRCUIT ENTRE UNE LIGNE Vdd ET UNE LIGNE Vss
Il y a destruction immédiate du circuit

Cette liste de pannes nous servira d'hypothèse pour le test des circuits CMOS, c'est à dire que l'on va chercher, soit à être capable de détecter chacun de ces défauts dès qu'ils surviennent, soit d'éviter qu'ils se produisent grâce à des précautions topologiques.

II.3 LES CIRCUITS AUTO-CONTROLABLES (SELF-CHECKING)

Le but de ce paragraphe est de donner les notions élémentaires qui ont permis d'établir la définition générale des circuits auto-contrôlables ou "self-checking".
Cette technique a l'avantage de couvrir des modes réels de défaillance qui sont actuellement bien étudiés, et elle ne nécessite aucun logiciel spécifique pour assurer le test en-ligne. La détection des erreurs est assurée par le matériel. La structure générale de ces circuits est donnée dans le schéma n°1. Les sorties du bloc fonctionnel sont codées et le contrôleur est utilisé pour vérifier les sorties. Dans le cas où le bloc fonctionnel génère un mot de sortie en dehors du code, le contrôleur produit une indication d'erreur. Couramment, les contrôleurs ont deux sorties codées en "double-rail" (10 ou

01 pour l'indication d'opération correcte et 11 ou 00 pour l'indication d'erreur). Des circuits complexes peuvent être conçus en réalisant les blocs de base dans la structure auto-contrôlable ou "self-checking". Dans ce cas, on utilise un contrôleur "double-rail" pour compacter les indications d'erreur des différents blocs en une indication d'erreur globale.

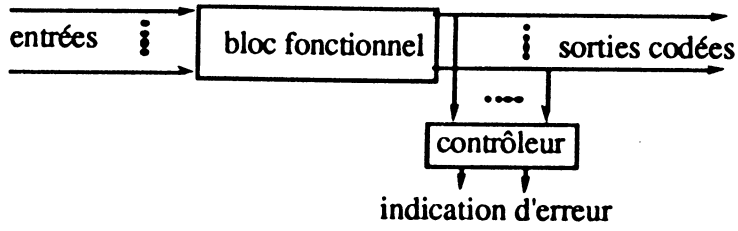


schéma n°1 structure générale des circuits "self checking"

Le but recherché par les circuits "self-checking" est appelé "totally self-checking goal" , il assure que la première sortie erronée du bloc fonctionnel est signalée par le contrôleur.

Pour assurer ce but, le circuit fonctionnel et le contrôleur doivent vérifier certaines propriétés qui sont formalisées au niveau mathématique .
Les conventions utilisées dans ce qui va suivre sont celles utilisées originellement par [SMI 78].

- Soit un circuit G combinatoire à r entrées primaires et q sorties primaires ,

$X = 2^r$ est l'ensemble des vecteurs d'entrée

$Y = 2^q$ est l'ensemble des vecteurs de sortie

$A \subset X$ est l'ensemble des vecteurs reçus lors du fonctionnement ou code d'entrée

$B \subset Y$ est l'ensemble des vecteurs émis lors du fonctionnement ou code de sortie

G(x,0) est la valeur de sortie d'une entrée x

G(x,f) est la valeur de sortie d'une entrée x en présence d'un défaut $f \in F$ ensemble des défauts.

Les définitions qui suivent sont dues à ANDERSON [AND 71]

Def 1 : G est "self-testing" ou auto-testable (ST) pour un ensemble F de défauts si :
$\forall f \in F, \exists a \in A$ tel que $G(a,f) \notin B$

Def 2 : G est "fault secure" (FS) pour un ensemble F de défauts si :
$\forall f \in F, \exists a \in A$ tel que l'on ait soit $G(a,f) = G(a,0)$ soit $G(a,f) \notin B$

Def 3 : G est "totally self checking" ou totalement auto-contrôlable (TSC) pour un ensemble F de défauts si :

Le circuit est "self-testing" et "fault secure" pour l'ensemble F.

A partir de la définition 3 on peut déduire la propriété suivante :

Un bloc fonctionnel G accomplit le totally self checking goal (TSC goal) si la première sortie erronée due aux défauts de l'ensemble F est en dehors du code de sortie.
Le TSC goal ne peut cependant être atteint que si l'hypothèse H1 est respectée.

Hyp 1 :
Entre l'occurrence de deux pannes quelconques de F il s'écoule un laps de temps suffisant pour que tous les vecteurs du code d'entrée soient appliqués aux entrées du circuit G

La plus grande classe de circuits fonctionnels capables d'assurer le TSC goal, en respectant l'hypothèse H1 est la classe des circuits "Strongly Fault Secure" [SMI 78], [DAV 78].

Def 4 : Un circuit G est "strongly fault secure" (SFS) pour un ensemble F de pannes si :
- a) soit le circuit est TSC
- b) soit le circuit est FS et si une nouvelle panne survient on retombe dans le cas a) ou b) pour la panne résultant de la combinaison des deux.

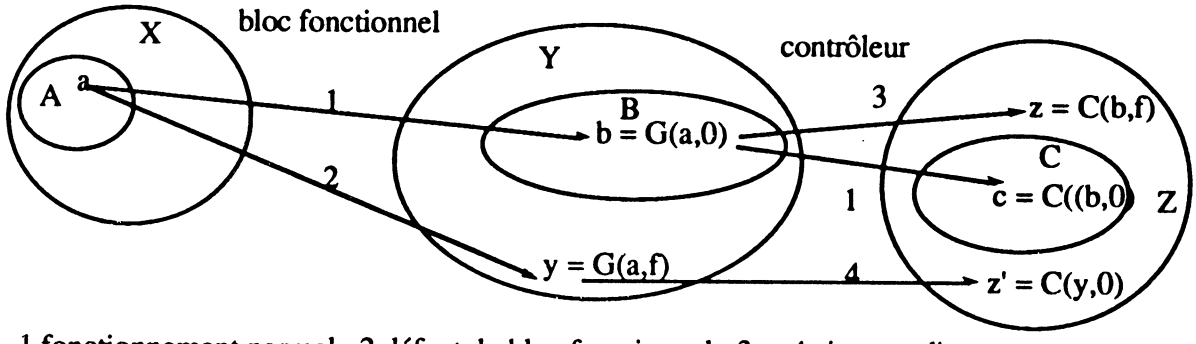
Def 5 : Un circuit G est à "codes disjoints" (CD) si :
$\forall a \in A,\ G(a,0) \in B,\ \forall x \in (X - A)\quad G(x,0) \notin B$

Def 6 : Un circuit G est un contrôleur "totally self-checking" s'il est totally self-checking et code disjoint.

En fait, la propriété "totally self-checking" n'est pas forcément nécessaire pour qu'un contrôleur assure sa mission dans la mesure ou celui-ci reste "code disjoint". A partir de cette observation, NICOLAIDIS [NIC 84] a introduit le concept de contrôleur "strongly code disjoint" (SCD).

Def 7 : Un circuit G est "strongly code disjoint" (SCD) pour un ensemble F de pannes si : avant l'occurrence d'une panne $f \in F$ le circuit G est CD et après l'occurrence d'une panne f on a
- a) soit G est ST
- b) soit G transpose les vecteurs d'entrée hors-code en vecteurs de sortie hors-code et si une nouvelle panne f' survient, la combinaison de pannes ff ' nous entraîne à nouveau vers le cas a) ou b).

Les contrôleurs "strongly code disjoint" représentent la plus large classe de contrôleurs qui, associés à des circuits fonctionnels "strongly fault secure" peuvent atteindre le "totally self-checking goal". Sous condition de respecter l'hypothèse H2 le schéma n°2 montre le fonctionnement d'une cellule de base d'un circuit qui atteint le "totally self-checking goal".



1 fonctionnement normal 2 défaut du bloc fonctionnel 3 et 4 signaux d'erreur

Schéma n°2 : Fonctionnement théorique d'un circuit autotestable

Hyp H2 :

Après l'occurrence d'une panne dans le contrôleur, il s'écoule un laps de temps suffisant pour que tous les vecteurs du code d'entrée A soient appliqués au bloc fonctionnel avant qu'une deuxième panne survienne dans le bloc fonctionnel ou dans le contrôleur.

Après l'occurrence d'une panne dans le bloc fonctionnel, il s'écoule un laps de temps suffisant pour que tous les vecteurs du code d'entrée B soient appliqués au contrôleur avant qu'une deuxième panne survienne dans le bloc fonctionnel ou dans le contrôleur.

L'hypothèse H2, appliquée à un contrôleur SCD nous permet de faire les constatations suivantes :

- Si une panne survient dans le bloc fonctionnel

soit elle ne modifie pas la fonction du circuit (panne latente)

soit elle provoque des erreurs qui seront en dehors du code de sortie B. Le contrôleur qui est à code disjoint fournira alors une indication d'erreur avant que la panne suivante se produise.

- Si une panne survient dans le contrôleur

soit le contrôleur reste à code disjoint (panne latente)

soit il existe au moins une valeur d'entrée du contrôleur qui provoquera un signal d'erreur avant que la panne suivante se produise.

II.4 CLASSIFICATION DES ERREURS ET CODES DETECTEURS

II.4.1 : Classification des erreurs

Les erreurs possibles peuvent être classifiées de la façon suivante :

- Les erreurs simples : Elles n'affectent qu'un seul bit dans un mot.
- Les erreurs unidirectionnelles : Elles affectent un nombre quelconque de bits d'un mot mais uniquement dans un seul sens (les erreurs sont toutes d'un seul type 0 -> 1 ou 1 -> 0 mais pas les deux à la fois).
- Les erreurs multiples : Elles modifient un nombre quelconque de bits dans les deux sens.

Les codes détecteur d'erreurs peuvent être classifiés de la manière suivante :

- Les codes séparables : Ce sont des codes dans lesquels les bits d'information et les bits de codage sont distincts et accolés (e.g . code de parité, code de Berger, duplication).
- Les codes non-séparables : Ce sont des codes dans lesquels les bits d'information et de codage sont confondus (e.g. m-parmi-n).

Les codes les plus souvent utilisés dans les circuits self-checking sont :

II.4.2 : Le code de parité

Ce code compte le nombre de "1" contenu dans le mot de k bits et affiche le résultat sur un bit supplémentaire. Si le nombre de "1" est pair, le bit de parité est égal à 1 (0 pour le code d'imparité). Si le nombre de "1" est impair, le bit de parité est égal à 0 (1 pour le code d'imparité). Il permet la détection des erreurs simples.

En pratique, pour contrôler ce code on calcule la parité en effectuant le OU-exclusif de tous les bits d'information et on la compare avec le bit de codage rajouté à l'information.

Il va de soi que le bloc fonctionnel doit être conçu de sorte que toutes les pannes ne provoquent que des erreurs simples sur les sorties primaires. Pour cela, il faut utiliser des règles de conception très précises au cours de la réalisation (voir paragraphe suivant).

II.4.3 : Le code de BERGER

Le code de Berger est un code non ordonné qui permet la détection des erreurs unidirectionnelles; il est obtenu par la concaténation des n bits de l'information I et de K bits de codage qui sont le complément du nombre de "1" contenu dans les bits d'information. K est égal à la borne supérieure entière de $Log_2(n+1)$

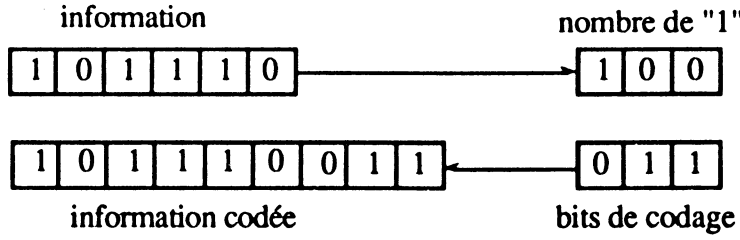


Schéma n° 3: Code de BERGER

Ce code est un code séparable optimal pour la détection des erreurs unidirectionnelles car c'est celui qui nécessite le moins de bits de contrôle pour un nombre donné de bits d'information.
Le principe d'un contrôleur de Berger est donné au schéma n°4, il est composé de deux éléments :

- Un générateur de code chargé, à partir des bits d'information, de régénérer les bits de contrôle complémentaires. Ce générateur est généralement conçu à partir d'additionneurs.
- Un contrôleur double-rail qui compare les bits de contrôle de l'information avec les bits de contrôle complémentaires délivrés par le générateur de code.

Le même principe peut être appliqué avec tous les codes séparables.

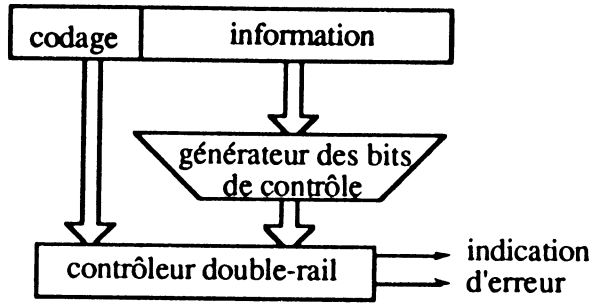


Schéma n° 4: Utilisation d'un code détecteur d'erreur

II.4.4 : Le code de duplication

Ce code est utilisé lorsque l'on veut détecter des erreurs multiples, il peut être de deux types :

- Le code de duplication (information + information)
- Le code double-rail (information + information complémentée)

En pratique, la duplication est obtenue en utilisant soit un bloc fonctionnel identique qui génère en parallèle la même information, soit un bloc fonctionnel dual qui délivre en parallèle le complément de l'information. Dans les deux cas le décodage est effectué en comparant les sorties à l'aide d'un contrôleur double-rail. La cellule de base d'un tel contrôleur est représentée par le schéma n°5 , pour obtenir un contrôleur possédant un nombre d'entrées plus important il suffit de mettre en cascade le nombre nécessaire de cellules de base .

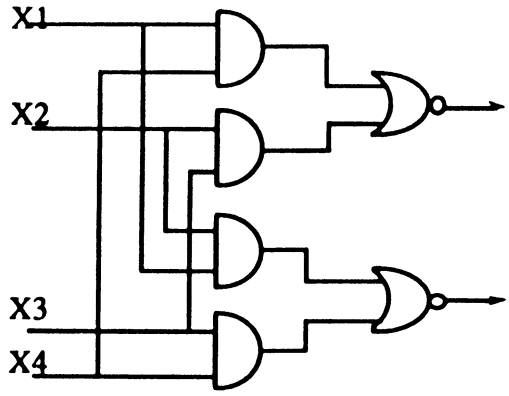


Schéma n°5 : Contrôleur double-rail

Toutes ces techniques de codage sont couramment employées pour le test intégré autonome. L'inconvénient majeur de ces méthodes provient de l'augmentation de surface qu'elles engendrent. Le schéma n°6 nous présente le pourcentage des bits supplémentaires pour les différents codes. On peut remarquer que cette augmentation est d'autant plus importante que l'on veut une protection plus grande, à moins d'effectuer un énorme effort de conception comme cela sera présenté dans le chapitre suivant.

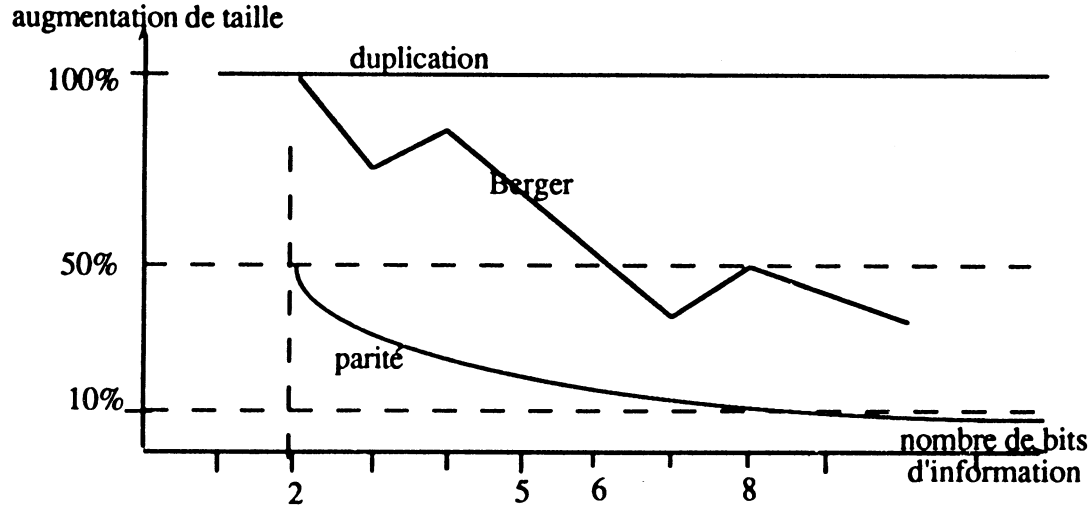


Schéma n° 6 : Evaluation de la surface ajoutée en fonction du code détecteur d'erreur choisi

II.5 REGLES DE CONCEPTION DES CIRCUITS SFS POUR LA CLASSE I

II.5.1 : Définitions

Avant de donner les règles de conception proprement dites nous allons rappeler quelques définitions qui permettront de mieux les comprendre

Définition d'un chemin

Un chemin est un ensemble ordonné de lignes entre deux points d'un bloc. Si dans l'ensemble < Ii1, ...,Iin> de lignes, la ligne Iij+1 est le successeur de Iij pour j = 1 jusqu'à n-1 alors cet ensemble ordonné est un chemin.

Définition d'un couple inversant

Un couple de lignes (Ii,Ij) sera appelé inversant, si une erreur du type D "niveau bas à la place du niveau haut" (respectivement $\overline{D}$ "niveau haut à la place du niveau bas") sur la ligne Ii ne peut produire qu'une erreur du type $\overline{D}$ (respectivement D) sur la ligne Ij.

Définition d'un couple non inversant

Un couple de lignes (Ii,Ij) sera appelé non inversant, si une erreur du type D "niveau bas à la place du niveau haut" (respectivement $\overline{D}$ "niveau haut à la place du niveau bas") sur la ligne Ii ne peut produire qu'une erreur du type D (respectivement $\overline{D}$) sur la ligne Ij.

Un couple (grille-drain) d'un MOS de signal est inversant.
Le couple (grille, drain/source) d'un transistor MOS utilisé comme interrupteur n'est pas défini vis à vis de la propriété d'inversion.
Tout autre couple est non inversant.

Définition de la parité d'inversion

La parité d'inversion Ip(P) d'un chemin P, dans lequel la propriété d'inversion est définie pour tous les couples de deux lignes successives, est le nombre binaire Ip(P) $\in$ {1,0} tel que :
Ip(P) = n modulo 2, où n est le nombre de couples inversants de lignes successives qui se trouvent sur le chemin P.

Définition du degré de divergence

Le degré de divergence d'une ligne Ii1 interne d'un bloc fonctionnel est le nombre de sorties primaires Iin du bloc pour lesquelles il existe des chemins < Ii1,.....,Iin>.

La conception d'un circuit" strongly fault secure" dépend du code de contrôle choisi, c'est à dire que les blocs seront conçus différemment suivant que l'on utilise le code de parité ou un code non ordonné (Berger , m parmi n). Chacun de ces codes n'est capable de contrôler que certains types de défauts (le code de parité ne détecte que les erreurs simples). Il faut donc que l'erreur produite en un point du circuit se propage jusqu'aux sorties primaires du circuit en erreur détectable par le code utilisé. Les règles exposées ci-dessous sont tirées d'un liste exhaustive décrite dans [NIC 84].

II.5.2 : Utilisation du code à parité (L'erreur interne après propagation jusqu'aux sorties doit donner une erreur simple)

R1 : le degré de divergence maximal du bloc fonctionnel pour l'ensemble de toutes les lignes du bloc est égal à 1

La règle R1 assure la propriété "fault secure" pour les défauts du groupe A et B(erreur simple aux sorties) à la condition qu'ils produisent des erreurs internes logiques. Il reste donc à se prémunir contre les erreurs B non logiques. Cette même règle assure la propriété "fault secure" pour les pannes du type A produisant des erreurs internes non logiques. Pour les pannes du type B générant des erreurs internes non logiques, la propriété "fault secure" n'est assurée que pour les circuits CMOS dynamique. Pour les circuits CMOS statiques, il sera nécessaire de disposer de nouvelles règles concernant l'éloignement des lignes connectées avec des sorties différentes, ceci correspond à la règle R3 exposée à la page suivante.

R2 : Une seule sortie primaire du bloc fonctionnel peut être erronée à cause d'une erreur multiple unidirectionnelle provoquée par un défaut du type C.

Dans la réalité, on utilise les règles R'2 et R"2 dont la mise en oeuvre est plus simple.

R'2 : Chaque ligne d'alimentation du bloc est utilisée pour alimenter des groupes de portes qui sont liés à une seule sortie primaire du bloc fonctionnel. (R'2 ===> R2).Voir schéma n°10.

R"2 : Une seule ligne d'alimentation est utilisée, l'extrêmité de cette ligne est utilisée pour alimenter un contrôleur. Si R"2 est respectée, alors les coupures des lignes d'alimentation peuvent être retirées des hypothèses de pannes.
Le schéma n°7 donne un aperçu d'une conception respectant les règles R1 et R'2.

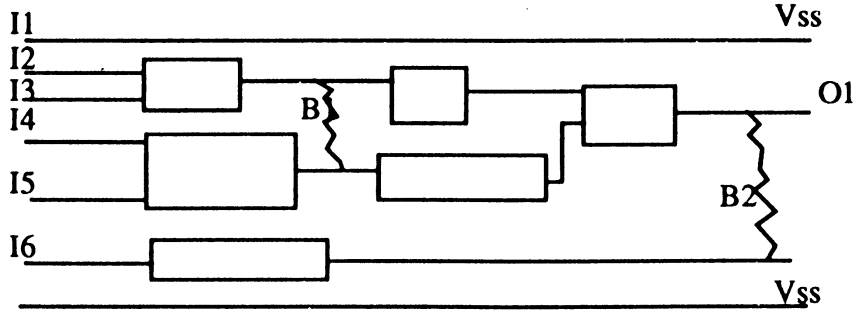


schéma n°7 : respect des règles R1 et R'2

Ces règles sont suffisantes pour assurer la propriété "fault secure", par contre pour assurer la propriété "strongly fault secure" il est nécessaire de disposer d'autres règles. Dans ce cas, il faut s'assurer qu'en présence d'une panne indétectable le circuit reste "fault secure", c'est à dire que les pannes indétectables ne doivent pas invalider les règles R1 et R2

A partir de là, on doit affiner les hypothèses de pannes, principalement la classe B qui se divise en deux; la classe B1 (courts-circuits entre 2 lignes reliées à la même sortie) et la classe B2 (courts-circuits entre deux lignes reliées à 2 sorties différentes) voir schéma n°7.

Les pannes du type B1 ne peuvent pas invalider les règles R1 et R2, par contre les pannes du type B2 vont transformer les 2 lignes court-circuitées en une seule ligne qui ne vérifiera plus la règle R1, il faut donc éliminer les pannes indétectables du type B2. Pour ne pas avoir à les chercher individuellement, il est préférable de toutes les éliminer. Pour cela, on peut soit utiliser la séparation topologique, soit utiliser des matériaux non court-circuitables, ceci correspond à la règle **R3**.

De même, il faut affiner l'hypothèse de panne D en deux catégories: D1 (courts-circuits entre deux lignes d'alimentation alimentant des portes liées à la même sortie) et D2 (courts-circuits entre deux lignes d'alimentation alimentant des portes liées à des sorties différentes).

Les pannes du type D2 sont susceptibles d'invalider la règle R2, il faut donc les éliminer soit en déplaçant les lignes, soit en utilisant des matériaux non court-circuitables. Ceci correspond à la règle **R4**. Le schéma n°8 donne une réalisation de la règle R4.

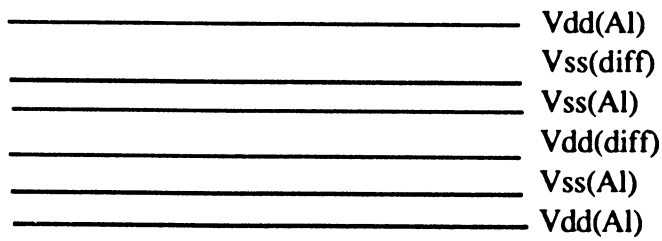


Schéma n°8 :Exemple d'organisation permettant de s'affranchir des défauts du type D2

PROPOSITION

Si des séquences < f1, f2, ... fk-1 > contenant des défauts du type B2 et pour lesquels le circuit est "strongly redundant" (voir schéma n° 9) ne peuvent pas survenir, et si des défauts du type D2 ne peuvent pas survenir non plus, alors le circuit conçu de façon à ce que son code de sortie détecte les erreurs simples et de façon à ce que les règles R1, R'2, R3, R4 soient vérifiées est "STRONGLY FAULT SECURE " pour la classe I.

Deux contre-exemples de cette proposition sont présentés par le schéma n°9.

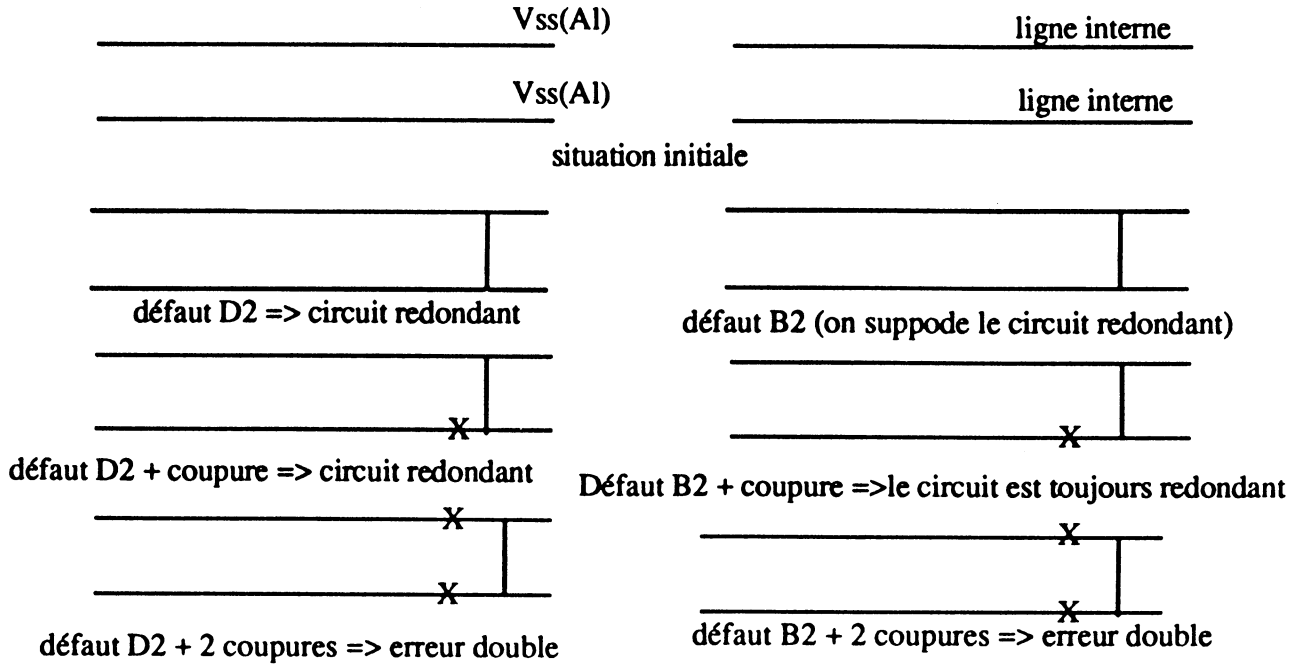


schéma n°9: la présence d'un défaut D2 ou B2 peut empêcher le circuit d'être SFS

II.5.3 : Utilisation d'un code non ordonné (l'erreur interne, après propagation jusqu'aux sorties, doit donner une erreur unidirectionnelle)

Règle R5 :

Tous les chemins entre une ligne divergente et les sorties primaires du circuit ont la même parité d'inversion.

La règle R5 assure la propagation des erreurs internes simples en erreurs unidirectionnelles aux sorties primaires du circuit et donc assure la propriété "fault secure" pour les défaut du type A et les défauts du groupe B ne donnant pas d'erreur double. Cette règle assure la propriété "fault secure" pour les pannes du type A produisant des erreurs internes non logiques [NIC 86] , [NAN 87]. La propriété "fault secure" est assurée pour les pannes du type B (B2') qui génèrent des erreurs internes non logiques que si on utilise des circuits CMOS dynamiques. On verra par la suite comment résoudre ce problème dans les circuits CMOS statiques à l'aide de la régle R7.

Règle R6 :

Pour assurer la propagation des erreurs multiples unidirectionnelles, provoquées par des défauts du type C en erreurs unidirectionnelles aux sorties primaires du circuit, tous les chemins entre les lignes d'occurrence de ces erreurs et les sorties primaires doivent avoir la même parité d'inversion.

Cette règle est très générale, en pratique elle est assurée par les règles R'6 et R"6.

Règle **R'6** :

Chaque ligne d'alimentation est utilisée pour alimenter des groupes de portes dont les sorties sont liées aux sorties primaires du bloc par l'intermédiaire de chemins qui ont la même parité d'inversion. (R'6 ===> R6)

Règle **R"6** :

Une seule ligne Vss et une seule ligne Vdd sont utilisées pour alimenter le bloc, les extrémités de ces lignes sont utilisées pour alimenter un contrôleur. Voir schéma n°10.

Ces règles assurent la propriété "fault secure" mais il est nécessaire de leur adjoindre d'autres règles pour assurer la propriété "strongly fault secure". Dans ce cas il faut s'assurer qu'en présence d'une panne indétectable le circuit reste "fault secure", c'est à dire que R5 et R6 ne sont pas invalidées.

Ainsi, comme précédemment, les hypothèses de pannes doivent être affinées. Les courts-circuits du type B et D doivent être divisés en deux sous catégories.

B'1 correspondant aux courts-circuits entre deux lignes ayant la même parité.

B'2 correspondant aux courts-circuits entre deux lignes ayant des parités d'inversion différentes

D'1 correspondant aux courts-circuits entre deux lignes d'alimentation affectées à des chemins ayant la même parité d'inversion.

D'2 correspondant aux courts-circuits entre deux lignes d'alimentation affectées à des chemins ayant des parités d'inversion différentes.

Les pannes du type B'1 ne peuvent pas invalider les règles R5 et R6, par contre les pannes du type B'2 vont transformer les deux lignes court-circuitées en une seule ligne qui ne vérifie plus la règle R5. Il faut donc éliminer les pannes indétectables du type B'2. Cela peut être réalisé en utilisant la règle R7.

Règle **R7** :

Pour éviter l'occurrence d'une panne multiple il faut éliminer les défauts du type B'2 soit par écartement topologique, soit par utilisation de matériaux non court-circuitables. (si non R7 alors non R5)

De même pour ne pas invalider la règle R6 il suffit d'appliquer la règle R8

Règle **R8** :

Les défauts du type D'2 doivent être éliminés de la même façon

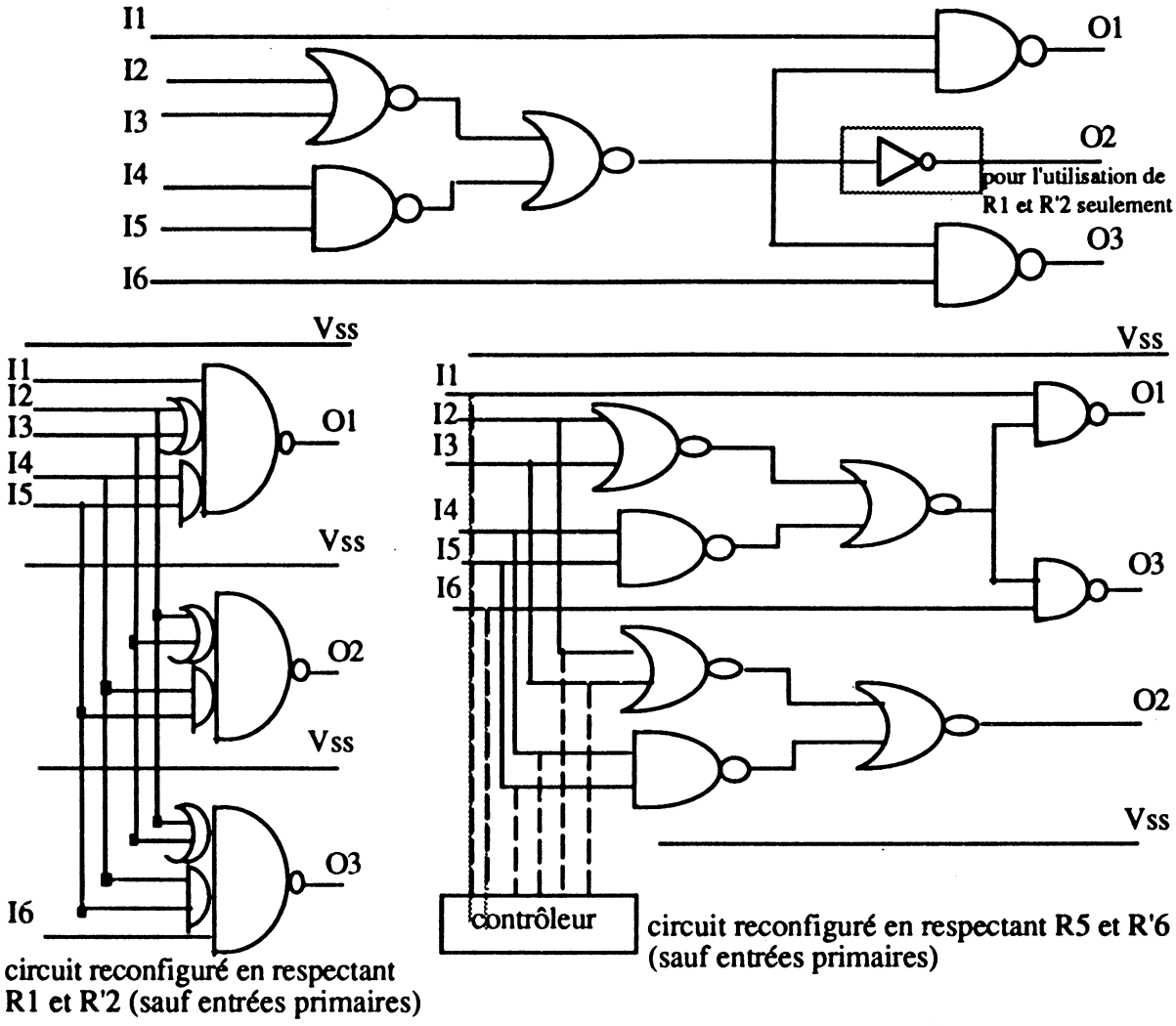


schéma n° 10 : exemple de modifications d'un circuit en fonction des règles R1 et R'2 et R5 et R'6

PROPOSITION

Si un circuit contrôlé par un code non ordonné a été dessiné en respectant les règles R5, R'6, R7, R8 alors ce circuit est "strongly fault secure" pour la classe I des hypothèses de pannes.

Le schéma n°11 donne comme exemple un PLA autotestable SFS.

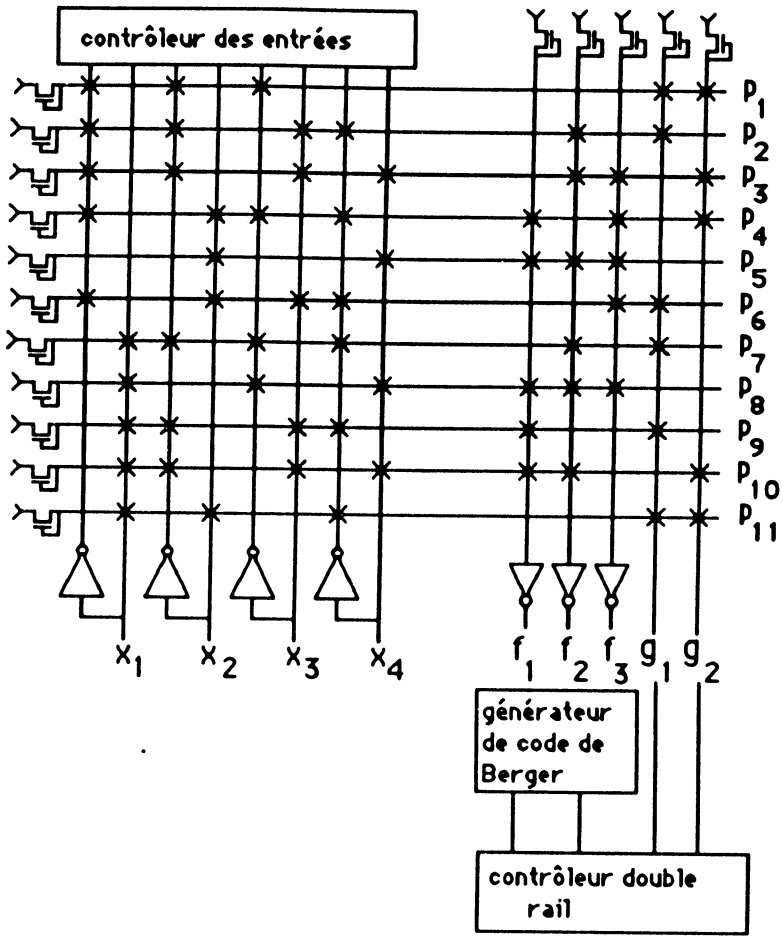


schéma n°11 : PLA autotestable

L'organisation interne d'un PLA nous assure que tous les chemins entre les lignes d'une même matrice et les sorties du circuit ont la même parité d'inversion, le circuit peut donc être testé; le contrôle des entrées primaires et un code non-ordonné.
g1 et g2 sont les sorties supplémentaires correspondant au codage de Berger, elles sont comparées avec le codage en Berger (recalculé en complémentaire) des sorties du circuit.

Nous avons proposé des règles assurant la propriété SFS pour les pannes simples. En présence d'erreurs internes logiques ou non logiques les circuits dessinés produiront, à leurs sorties, des erreurs simples ou unidirectionnelles (selon les règles choisies) détectables par le code utilisé. Cependant certaines erreurs présentes aux sorties du circuit peuvent apparaître avec un niveau indéterminé. Dans ce cas, le problème suivant peut apparaître :

en se référant au schéma n° 12, on se peut trouver dans une situation où le contrôleur I interprète la valeur indéterminée aux sorties du bloc fonctionnel I comme si s'était la valeur correcte (ce qui empêche la détection) tandis que le bloc I + 1 interprète cet état comme un état erroné. Le système fournira donc un état erroné sans qu'aucune détection n'ait lieu

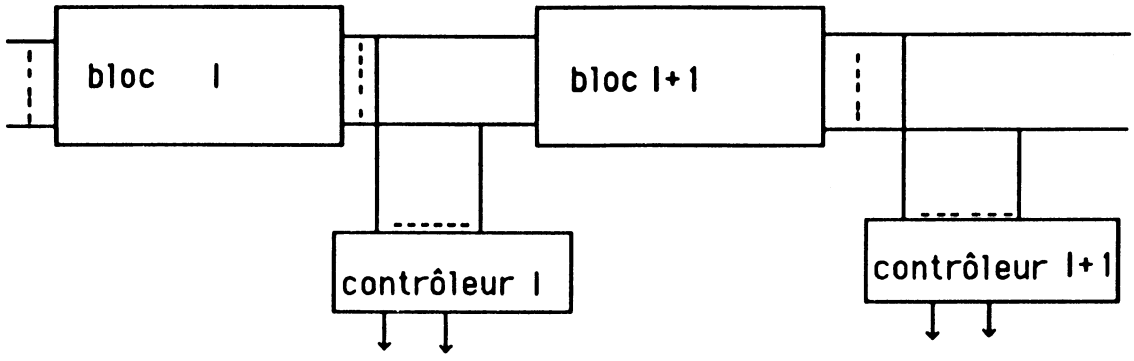


Schéma n° 12 : Système self-checking

On peut par ailleurs remarquer que toutes les erreurs produites par les pannes d'un quelconque bloc et donnant un niveau indéterminé seront redressées par le registre de sortie de ce bloc. Par conséquent, seules les pannes survenant dans le registre même peuvent poser le problème décrit ci-dessus. On peut cependant affirmer que ces pannes représentent un faible pourcentage du nombre total des pannes . De toute façon, ces pannes seront détectées car le MAPS sera conçu par deux processeurs "self-checking". Seule la couverture des pannes multiples affectant simultanément deux registres de sortie des deux processeurs sera faiblement affectée par ce problème.

Le tableau n° 2 résume les règles de dessin à utiliser en fonction de la propriété que l'on veut obtenir pour le circuit.

hypothèses de panne	propriétés	erreurs détectées par le code de sortie	
		simples	unidirectionnelles
défauts simples sauf rupture d'une ligne d'alimentation et courts-circuits	FS	R1	R5
	SFS	R1	R5
défauts simples sauf courts-circuits	FS	R1, R2	R5, R6
	SFS	R1, R2	R5, R6
défauts simples sauf courts-circuits entre 2 lignes d' alimentation	FS	R1, R2	R5, R6
	SFS	R1, R2, R3	R5, R6, R7
défauts simples sauf certains courts-circuits	FS	R1, R2	R5, R6
	SFS	R1, R2, R3, R4	R5, R6, R7, R8

Tableau n°2 : Règles de dessin pour des hypothèses de pannes au niveau des transistors

II.6 INTERCONNEXION DE BLOCS COMBINATOIRES SFS

Pour qu'un circuit combinatoire, composé de plusieurs blocs SFS soit lui même SFS il faut que cette propriété soit assurée au niveau des interconnexions. Les règles de conception qui en découlent s'appliquent lorsque l'on utilise des codes détecteurs d'erreur capables de mettre en évidence des pannes simples ou unidirectionnelles mais qui ne font pas appel à une duplication . La conception de systèmes complexes capables de détecter des erreurs multiples fait appel à la duplication classique de chacun des blocs.

1er cas : Blocs fonctionnels SFS et SCD

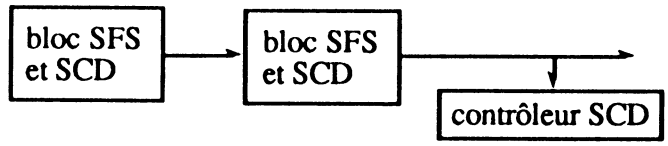


Schéma n° 13 : Blocs fonctionnels SFS/SCD et contrôleur SCD

La propriété SFS et SCD de chaque bloc fonctionnel assure que toute erreur sur une sortie d'un des éléments sera propagée en un vecteur hors-code sur les sorties primaires du système. Cependant la conception d'une fonction complexe à la fois "strongly code disjoint" et " strongly fault secure" est souvent difficilement réalisable [NAN 89].

2ième cas : blocs fonctionnels SFS

-a) Vérification par un contrôleur unique

La propriété "strongly fault secure" de l'ensemble est assurée si les blocs respectent les règles de propagation suivante :

-Chaque erreur simple (respectivement unidirectionnelle) aux entrées d'un bloc qui vérifie la règle R1 ou R5 (respectivement R5) est propagée jusqu'aux sorties primaires de ce bloc soit comme une erreur simple soit comme une erreur unidirectionnelle (respectivement comme une erreur unidirectionnelle).

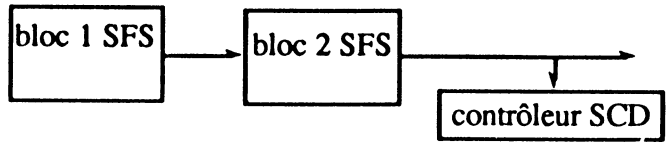


Schéma n° 14: Blocs fonctionnels SFS et contrôleur SCD

-b) Vérification par plusieurs contrôleurs (Voir schéma n°15)

Cette méthode permet d'assembler deux blocs "strongly fault secure" sachant que le second bloc est incapable de détecter la présence d'erreurs simples (ou unidirectionnelle) sur ses entrées. Le second contrôleur, situé entre les deux blocs, doit être placé de telle sorte qu'il puisse vérifier les connexions jusqu'à certains points de bifurcation (points critiques) [NIC 84] à partir desquels le second bloc est capable de détecter toute présence de panne.

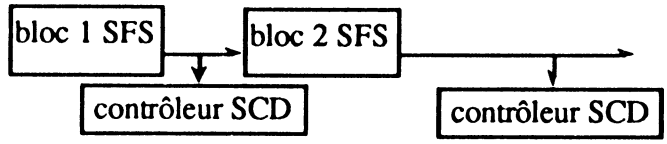


Schéma n°15 : Blocs fonctionnels SFS et contrôleurs SCD

II .7 LES BLOCS STRONGLY CODE DISJOINT

La réalisation concrête d'un contrôleur strongly code disjoint est assez délicate et généralement possible que pour un modèle restreint de pannes. Cependant, la technique UBIST que nous détaillerons par la suite permettra d'obtenir cette propriété beaucoup plus facilement par application de vecteurs du code et de vecteurs hors-code. En conséquence, nous ne donnerons pas dans ce document les règles de conception des contrôleurs.

II.8 CONCLUSION

Nous avons vu dans ce chapitre les principales techniques de test utilisées pour la conception de circuits intégrés qui ont la faculté de détecter leurs propres erreurs immédiatement après l'apparition du premier résultat erroné. Cependant, les méthodes d'autotest en-ligne ne sont effectives que si certaines conditions sont respectées (e.g hypothèses H1 et H2). Dans le cas de réalisations pratiques de circuits, le respect de ces contraintes devient extrêmement difficile, voire impossible à satisfaire.

C'est donc pour résoudre ce problème que l'on va introduire la technique UBIST en combinant les avantages de la technique self-checking que l'on vient de présenter avec la technique BIST. Nous obtiendrons ainsi une grande souplesse d'utilisation ainsi qu'une couverture de panne très élevée.

CHAPITRE III

BIST UNIFIE

III.1 INTRODUCTION

Jusqu'à présent, nous avons présenté des techniques qui permettaient, soit d'effectuer un test hors-ligne (BIST), soit un test en-ligne (circuits auto-contrôlables), mais ces techniques sont indépendantes les unes des autres. C'est donc en toute logique que l'on a cherché à intégrer conjointement ces deux techniques pour aboutir à l'étape ultime du test intégré qu'est la technique UBIST (unified built-in self test) [NIC 90b], [NIC 88]. Cette méthode, née des limitations de la théorie des circuits auto-contrôlables, permet d'exploiter au maximum les avantages que chacune des deux techniques peut apporter à l'autre.

Ce chapitre va donc être essentiellement consacré aux rappels et aux définitions qui ont permis à Nicolaidis d'élaborer la technique UBIST.

III.2 LIMITATIONS DES CIRCUITS AUTOTESTABLES

La théorie des circuits autotestables est basée sur un certain nombre de limitations qui ne permettent pas d'exploiter facilement un de ces circuits dans un système complexe. Ces limitations peuvent être énumérées de la manière suivante

1) La propriété SCD ou TSC du contrôleur ne peut être atteinte que si le bloc fonctionnel auquel il est rattaché est capable de générer un ensemble minimal de vecteurs lui permettant d'assurer le test. Cette condition est très rarement vérifiée dans les circuits réels. De plus, la conception de contrôleur SCD n'est possible que si l'on utilise un modèle restreint de pannes (collage logique).

2) Un système conçu avec des contrôleurs SCD et des blocs SFS doit recevoir, de façon régulière, l'ensemble des vecteurs du code d'entrée pour assurer le TSC goal. Cela ne peut être réalisé qu'au prix de restrictions du logiciel à utiliser.

3) Dans les circuits autotestables, les contrôleurs et les blocs fonctionnels sont conçus pour couvrir les pannes simples, mais, pendant la fabrication, des pannes multiples apparaissent dans certains circuits. Il faut donc s'assurer que toute panne multiple a été détectée avant d'utiliser le circuit. Certaines pannes multiples des contrôleurs ne peuvent d'ailleurs être détectées que par l'usage de mots hors-code.

La technique UBIST développée par NICOLAIDIS permet de s'affranchir de ces limitations, pour cela elle s'appuie sur le concept de contrôleurs auto-activables ou "self-exercising" .

III.3 LES CONTROLEURS AUTO-ACTIVABLES OU SELF EXERCISING

L'utilisation de contrôleurs auto-activables permet d'assurer la propriété SCD sans avoir besoin de respecter des contraintes de conception. Ceci est réalisable grâce à l'intégration d'un générateur de test qui est déclenché régulièrement au cours d'un phase de test insérée dans le déroulement du programme.

Ainsi, d'une part tous les vecteurs qui appartiennent au code de sortie du bloc fonctionnel sont obligatoirement générés et, d'autre part, une partie, ou même tous les vecteurs hors-code peuvent être injectés sur les entrées. Une circuiterie adéquate (CNCI ou code non-code indicator) indique la nature (code ou hors-code) de chaque vecteur injecté. Ceci est nécessaire pour assurer la consistance des indications d'erreur (voir schéma n° 1)

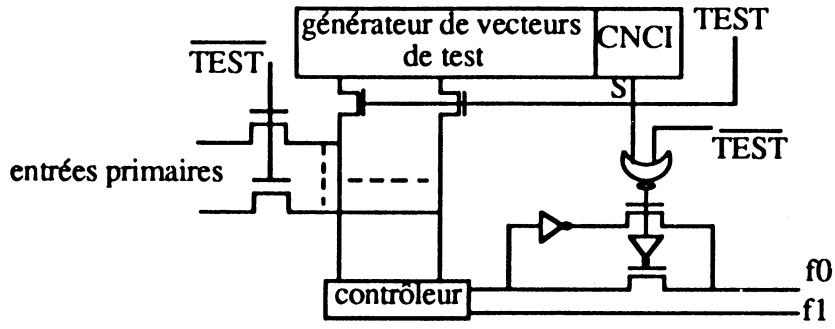


Schéma n°1 : Description du fonctionnement d'un contrôleur auto-activable ou "self-exercising"

Les contrôleurs auto-activables ont des structures et des comportements différents des contrôleurs classiques, il est donc nécessaire de redéfinir leurs propriétés.

Définition

Un contrôleur auto-activable est auto-testable pour un ensemble de pannes F (F est l'ensemble des pannes survenant dans le contrôleur, le bloc CNCI et le générateur de test) si pour tout f appartenant à F

- Soit un vecteur du code d'entrée est transformé en un vecteur hors-code de sortie pendant le fonctionnement du circuit.
- Soit un vecteur hors-code est produit aux sorties du circuit pendant la phase de test.

Définition

Un contrôleur auto-activable est SCD si:

avant l'apparition d'une panne quelconque , il est à code disjoint et si pour chaque panne appartenant à l'ensemble F soit:

- 1) le contrôleur auto-activable est auto-testable soit
- 2) le contrôleur transpose toujours un mot hors-code appliqué sur ses entrées en un mot hors-code sur ses sorties et si une nouvelle panne, appartenant à F, survient, la combinaison des deux pannes nous donne encore le cas a) ou le cas b).

Suite à ces définitions, on peut énoncer le théorème I qui a été démontré dans [NIC 88a].

Théorème I

Un contrôleur "self-exercising" qui reçoit tous les vecteurs d'entrée hors-code est SCD

- Pour tous les types de pannes possibles affectant le contrôleur (sauf celles qui introduisent un comportement séquentiel)
- Pour toutes les conceptions possibles de contrôleur
- Indépendamment du code de sortie du bloc fonctionnel et du programme utilisateur
- Pour tous les types de pannes affectant le bloc CNCI
- Pour tous les types de pannes affectant le générateur sauf celles pour lesquelles les deux conditions suivantes sont toutes les deux vérifiées

1) L'ordre des mots du code et hors-code n'est pas modifié

2) Certains mots hors-code ne sont pas générés

III.4 PRINCIPE DE LA CONCEPTION UBIST

La conception en vue du test unifié (unified built-in self test) assure la détection des pannes du circuit, pendant la phase de fonctionnement normal (test en ligne) et au cours de la phase de test hors-ligne, en déclenchant la génération interne de séquences de vecteurs de test qui vérifient le bon fonctionnement des blocs fonctionnels et des contrôleurs.

Les mécanismes capables de délivrer les séquences de vecteurs de test hors-ligne sont réalisés à l'aide de BILBO [KON 79]. Ces éléments ont plusieurs modes de fonctionnement (registre classique, registre à décalage, registre à décalage et à rebouclage linéaire) [FUJ 85]. Par contre, dans la technique UBIST, les BILBO subissent quelques modifications pour devenir des UBILBOs, c'est à dire qu'ils sont composés d'un BILBO et d'un circuit du type CNCI, ou circuit indicateur de la nature du vecteur (voir schéma n°1).

Pendant le fonctionnement normal du circuit, le test en ligne permet de vérifier les blocs fonctionnels SFS et les contrôleurs SCD.

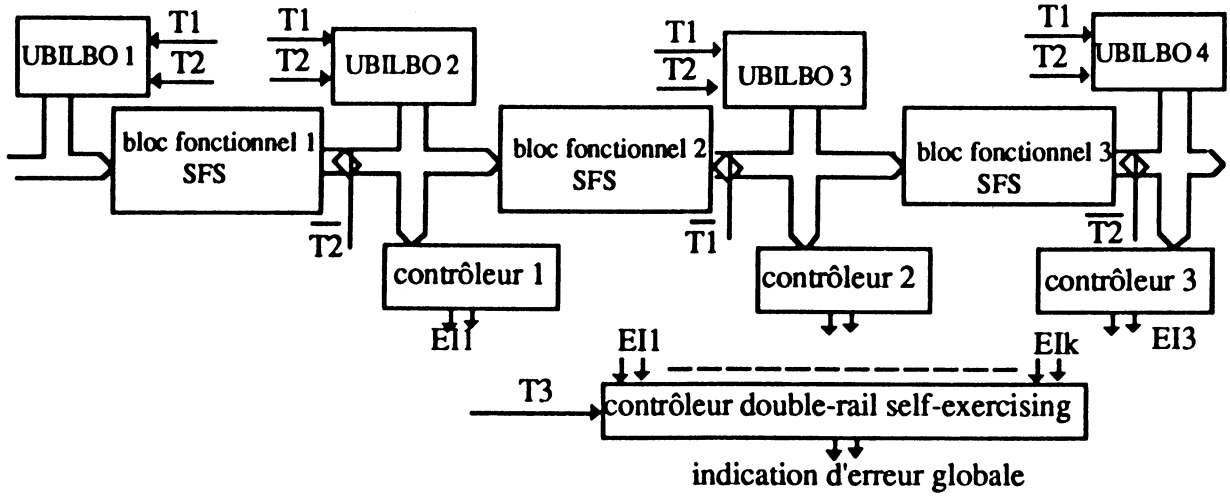


Schéma n°2 : Organisation générale de la technique UBIST

Lors du test hors-ligne, le bon fonctionnement du contrôleur double-rail final qui compacte les signaux d'indication d'erreur intermédiaire (EI1 .....EIk) est vérifié durant la phase "T3" car ce contrôleur dispose de son propre générateur.

Lors de la phase "T1", les UBILBOs impairs (1, 3 ...) délivrent des vecteurs de test sur les entrées des blocs fonctionnels impairs (1, 3 ...) et sur les entrées des contrôleurs pairs (2, 4 ...). Les réponses de chacun des blocs sous test (1, 3 ...) sont vérifiées par les contrôleurs impairs correspondants, mais elles sont aussi compactées en vue d'obtenir une signature finale par les UBILBOs pairs associés (2, 4 ...) Pendant cette phase, la défaillance d'un contrôleur intermédiaire pair est alors immédiatement signalée par le contrôleur double-rail final.

Dès que la phase de test "T1" est terminée, les signatures des différents UBILBOs pairs sont contrôlées. En général, on initialise les LFSRs des UBILBOs de telle sorte que la signature compactée soit du type 0101010..... et l'on connecte en série les analyseurs de signature au contrôleur double-rail final. Un flip-flop, lui aussi connecté au contrôleur double-rail final, est associé à cette signature de telle sorte qu'il délivre à chaque coup d'horloge la séquence complémentaire du type 1010101..... Toute erreur de signature sera donc signalée par le contrôleur double-rail final.

On applique le même principe lors de la phase de test "T2" aux blocs fonctionnels pairs et aux contrôleurs impairs.

III.5 CONCLUSION

La technique UBIST assure :

- La propriété SFS des blocs fonctionnels pour toutes les pannes simples
- La propriété SCD des contrôleurs quel que soit le type de panne considéré (Théorème I)
- L'application régulière d'un ensemble suffisant de vecteurs pour assurer la vérification des blocs fonctionnels et des contrôleurs afin d'éviter la latence des pannes.

D'autre part la technique assurant le test hors-ligne du MAPS permet d'obtenir :

- La détection de toutes les pannes simples et la plupart des pannes multiples affectant les blocs fonctionnels
- La détection de toutes les pannes qui surviennent dans les contrôleurs et les UBILBOs

Si l'on considère comme hypothèse que les circuits défectueux en fin de fabrication ont été éliminés (ceci est assuré par la très grande couverture de panne du test hors-ligne décrite ci-dessus), on peut supposer que l'apparition d'une panne lors de la durée de vie du circuit est très faible et que l'apparition simultanée d'au moins deux pannes capables de masquer une erreur est négligeable.
Une telle méthode de conception nous permet donc d'obtenir le TSC goal pour un circuit intégré.

Le dernier point important concerne les pannes transitoires.
Les circuits self-checking ne sont pas capables de différencier ces pannes par rapport aux pannes permanentes. Sachant que les pannes transitoires sont beaucoup plus fréquentes que les pannes permanentes, il est très intéressant de pouvoir lancer un test hors-ligne pour déterminer le type de panne que le circuit vient de rencontrer pour continuer à utiliser le système si la panne était transitoire. De cette façon la disponibilité du système est augmentée de façon significative.

CHAPITRE IV

LES SYSTEMES FAIL-SAFE

IV.1 INTRODUCTION

Les circuits intégrés autotestables que nous avons présentés dans le chapitre précédent ne peuvent pas être utilisés pour commander des systèmes électromécaniques utilisés dans la plupart des installations critiques, comme celles qui sont utilisées dans l'industrie à haut risque (chimie, nucléaire...) ou les transports, et notamment le chemin de fer. Seuls les systèmes "fail-safe" conventionnels [MIN 67], [TAK 71], ou systèmes ne donnant que des sorties correctes ou sûres, sont capables d'accomplir cette tâche car ils génèrent des signaux qui sont "fail-safe" individuellement. Malheureusement, ces systèmes, tels qu'il sont définis, ne peuvent pas être intégrés.

Dans ce chapitre nous allons donc rappeler les définitions fondamentales des circuits "fail-safe" revues et généralisées dans [NIC 89] , faire une liaison avec les circuits auto-contrôlables et introduire le concept de circuit "strongly fail-safe".

IV.2 DEFINITIONS DE BASE

Définition D0 :

Un circuit G à une seule sortie primaire est "0" fail-safe (respectivement "1" fail-safe) pour un ensemble de vecteurs d'entrée X et pour un ensemble de pannes F si :
$\forall x \in X, \forall f \in F : G(x,f) = G(x,0)$ ou $G(x,f) =$"0" (respectivement "1")
G(x,0) correspond à la fonction correcte et G(x,f) à la fonction défaillante, "0" (respectivement "1") représente l'état sûr et "1" (respectivement "0") l'état non sûr.

Cependant, ce concept peut être étendu à un système comportant un nombre d'états quelconques. Dans ce cas, l'ensemble des sorties Y peut être divisé en deux sous-ensembles, Os l'ensemble des états sûrs et On l'ensemble des états non sûrs. Os comporte les états qui ne créent jamais de situation dangereuse, même s'ils surviennent par erreur. Ces nouvelles hypothèses nous permettent d'établir la définition suivante.

Définition D1:

Un système G est fail-safe pour un ensemble de pannes F, un ensemble de vecteurs d'entrée X et un ensemble d'états sûrs Os si
$\forall x \in X, \forall f \in F : G(x,f) = G(x,0)$ ou $G(x,f) \in Os$

Dès à présent nous pouvons remarquer qu'un circuit "fault secure" conçu de telle sorte que l'occurrence de vecteurs de sortie erronés ne provoque pas de situation dangereuse peut être considéré comme un circuit "fail-safe".

IV.3 LE TOTALLY FAIL-SAFE GOAL ET LES CIRCUITS STRONGLY FAIL-SAFE

Si un ensemble G est "fail safe" pour un ensemble de pannes F, et si l'occurrence d'une première panne f1 appartenant à F ne génère pas de vecteur de sortie erroné non sûr, alors la sécurité est assurée. Cependant, le vecteur sûr ainsi généré peut appartenir à l'ensemble des vecteurs sûrs obtenu en fonctionnement normal. Dans ce cas, la première panne n'est pas signalée, puisque de manière générale, les systèmes "fail-safe" ne possèdent pas nécessairement de moyens de détection. Si le système fonctionne pendant un temps assez long, une seconde panne f2 appartenant à F peut apparaître et la panne double (combinaison de f1 et f2) créée peut alors ne plus appartenir à l'ensemble F.

La propriété "fail-safe" est alors perdue. Il est donc essentiel de détecter une panne dès son apparition et empêcher le système défaillant de fonctionner. La détection des erreurs s'inspire des propriétés qui avaient été établies pour les circuits "self-checking", notamment grâce aux définitions suivantes. Si un système ne possède aucun moyen de détection des erreurs, il suffit de lui rajouter un mode de test du type hors-ligne.

Définition

Un circuit est "self-testing" pour un ensemble de pannes F, si pour chaque f appartenant à F il existe au moins un mode durant lequel la panne est détectée.

Définition

Un circuit est "totally fail-safe" s'il est à la fois "fail-safe" et "self-testing".
Un circuit "totally fail-safe" atteind le "totally fail-safe goal" si l'hypothèse H1 est vérifiée.

Hypothèse H1

- 1) Les pannes apparaissent une à une.

-2) Entre l'occurrence de deux pannes successives, il s'écoule un laps de temps assez long pour que le circuit soit contrôlé avec les moyens mis en oeuvre pendant les différents modes de fonctionnement possibles.

Si l'hypothèse H1 est vérifiée, la plus grande classe de circuits "fail-safe" capables d'atteindre le "totally fail-safe goal" est celle des circuits "strongly fail-safe".

Définition

Un circuit G est "strongly fail-safe" pour un ensemble de pannes F,si pour tout f appartenant à F

-1) G est "totally fail-safe" ou

-2) G est "fail-safe" et si une nouvelle panne f2 appartenant à F survient, la panne double ainsi créée redonne le cas 1) ou le cas 2).

Si on ne dispose pas de moyen de détection, la condition nécessaire et suffisante pour que le système soit "strongly fail safe" est qu'il doive être "fail-safe" pour l'ensemble des pannes multiples. Les circuits "fail-safe" décrits dans la littérature sont généralement conçus en utilisant des composants dupliqués (redondance) [MIN 67] ou des composants dont le MTBF est très élevé pour les pannes considérées comme "dangereuses", c'est la technique d'évitement des pannes [FUT 88]. En fait, la propriété "fail-safe" ne peut pas être assurée pour des pannes multiples lorsque l'on utilise la redondance à cause de l'apparition possible de pannes doubles dans deux composants dupliqués. Cela correspondrait à la présence de sorties erronées non sûres.
La technique d'évitement n'est pas non plus envisageable car certaines pannes telles que les s-on et les s-open ont des conséquences contraires, ce qui pourrait produire un état erroné non-sûr. De plus, la probabilité d'occurrence de telles pannes ne peut pas être considérée comme négligeable. Ces techniques ne sont donc pas compatibles avec l'utilisation de systèmes intégrés VLSI "strongly fail-safe".

En conséquence,nous allons nous intéresser à la conception de systèmes "fail-safe" utilisant une technique de codage (circuit self-checking) et une interface qui sera "fail-safe" grâce à l'intégration de composants "fail-safe" et même "strongly fail-safe" puisque l'on utilise la technique BIST qui assurera la détection de la première panne.

IV.4 TRANSFORMATION D'UN CIRCUIT SELF-CHECKING EN UN CIRCUIT FAIL-SAFE

De manière générale les circuits "fault secure", qui sont un sous-ensemble des circuits "fail-safe", ne peuvent pas remplacer efficacement les circuits "fail-safe". En effet, certains processus critiques demandent que chaque signal de sortie d'une fonction soit "fail-safe" individuellement. Pour cela le micro-contrôleur MAPS génère des sorties codées en fréquence.
Cette technique consiste à faire correspondre à un état non sûr le passage d'une fréquence (état 1) et à considérer tout autre état électrique (état 0) comme état sûr (notamment le blocage du passage de la fréquence). Cette solution a aussi l'avantage de s'adapter à la commande des éléments électromécaniques (actionneurs) qui se trouvent généralement placés en bout de chaîne des fonctions critiques [SAL 75], [FUT 88].

Le schéma n°1 présente le principe de réalisation de l'interface de sortie d'un système "self-checking" de telle sorte que le système global (circuit + interface) respecte toujours la définition D0.
Dans le schéma n°1 les blocs fonctionnels sont "strongly fault secure" et les contrôleurs "strongly code disjoint". Le circuit délivre n sorties (I1 ..... In) fonctionnelles codées avec k bits (C1 ..... Ck). Les signaux d'indication d'erreur intermédiaires sont interprètés par le contrôleur double-rail final pour donner une indication d'erreur globale (f1,f2).
Comme les blocs fonctionnels sont SFS et l'ensemble des contrôleurs est SCD, chaque vecteur erroné sera donc un vecteur hors-code.
Si le vecteur (I1.....In,f1,f2) appartient au code, alors l'interface fournit une fréquence Fe correspondant à l'état non sûr (état 1) aux sorties (O1 .....On) en fonction de l'autorisation de passage délivré par le vecteur (I1 .....In).

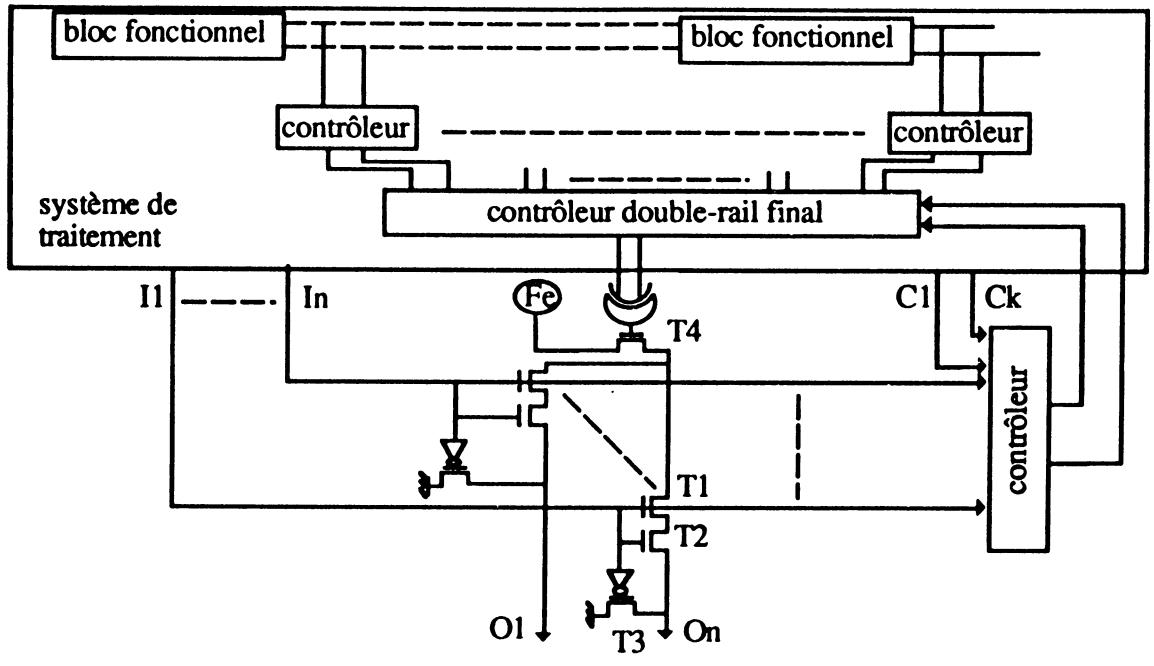


Schéma n°1 : Principe d'une interface "fail-safe"

La propriété "fail-safe" de l'interface est assurée en dupliquant certains transistors (T1 et T2) mais comme par hypothèse seule l'occurrence d'une panne simple est possible, il n'est pas nécessaire de dupliquer la porte XOR en sortie du contrôleur double-rail final.
La propriété "strongly fail-safe" de l'interface est assurée en utilisant la technique BIST, c'est à dire en imposant périodiquement des entrées issues d'un générateur de test (non représenté sur le schéma n°1) en remplacement de celle provenant du système de traitement.
Cette technique peut aussi être appliquée dans le cas du code double-rail comme cela sera décrit par la suite pour le cas particulier du micro-contrôleur MAPS.

En conclusion nous pouvons remarquer qu'une telle architecture composée de deux systèmes (interface + circuit self-checking) chacun "fail-safe" qui place les sorties dans un état sûr dès qu'un vecteur hors-code est généré forme un système global "fail-safe". La démonstration de cette conclusion a été établie dans [NIC 89].

IV.5 CONCLUSION

Les principes de base rappelés dans ce chapitre seront directement utilisés pour concevoir les interfaces "fail-safe" du circuit MAPS. L'interface de sortie du MAPS reprendra directement le principe de l'interface "fail-safe" présentée au schéma n°1 en l'adaptant à l'architecture interne du circuit (duplication). Nous verrons aussi que les critères de sécurité appliqués aux transports nous imposent l'élaboration d'interfaces "strongly fail-safe" et l'utilisation de plusieurs fréquences de sortie.

CHAPITRE V

CONCEPTION DU MAPS

V.1 INTRODUCTION

Ce chapitre va nous permettre, grâce à la description d'un réseau ferroviaire établie au chapitre I, de présenter de manière détaillée l'architecture interne du MAPS. Cette architecture traite les problèmes demandés et répond ainsi aux besoins spécifiques du client .

On verra donc, au fil du chapitre, quels sont les organes essentiels à intégrer, tant au niveau de la carte qu'au niveau du circuit MAPS pour traduire matériellement les spécifications établies. Cependant, l'architecture interne ne sera décrite que fonctionnellement sans faire intervenir le test qui ne sera abordé qu'au chapitre suivant.

V.2 SPECIFICATIONS EXTERNES

V.2.1 : La carte

Le circuit MAPS doit pouvoir communiquer avec le poste d'aiguillage informatisé où se trouve l'ordinateur central pour l'informer de l'état du réseau et recevoir ses instructions. C'est aussi par l'intermédiaire du poste d'aiguillage que les MAPS pourront communiquer entre eux.
Le circuit MAPS commande en fréquence les divers actionneurs décrits au chapitre 1 et reçoit comme entrée le compte-rendu de leur état, après l'envoi d'une commande. Deux de ces entrées, reliées à des pédales électroniques, sont des entrées d'informations utilisées pour compter les essieux et indiquer leurs sens.
Le circuit MAPS a aussi besoin d'accéder à une ROM externe qui contient des constantes nécessaires au décodage des messages reçus ainsi que le ou les programmes d'application. La seconde ROM (ROM*) contient les informations duales.

L'étude menée par la CSEE a établi la liste suivante des broches du circuit :
- 8 sorties fréquence fonctionnelle, plus 1 sortie fréquence de contrôle de l'alimentation.
- 8 entrées de compte-rendu dont 2 plus spécifiques (comptage) reliées à un automate interne.
- Une ROM externe de 8 Koctets.
- 2 ou 4 plots d'alimentation
- Une horloge

- La communication entre le poste d'aiguillage et le circuit se fait par l'intermédiaire d'un microprocesseur de communication externe qui gère les problèmes de protocole et une RAM interne double-accès.

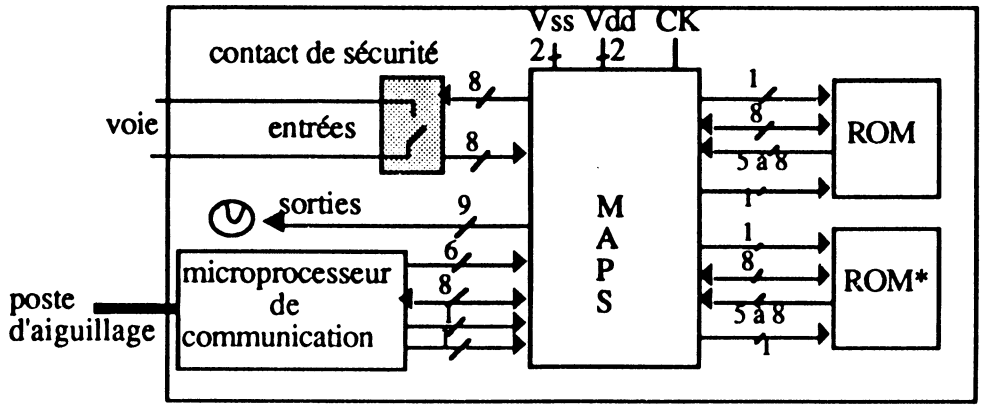


schéma n°1 : La carte

V.2.2 : Liaison avec la ROM externe

La ROM externe est reliée au micro-contrôleur MAPS par un bus qui assure une communication bidirectionnelle (adresse sur $\phi 2$ et données sur $\phi 3$) sur 8 bits et une communication unidirectionnelle sur 8 bits. Le compteur ordinal, inscrit sur 2 octets inclus dans la RAM interne de travail (voir schéma n°2), pourrait adresser une ROM de 64 Koctets, cependant les spécifications établies par la CSEE prévoient l'usage d'une ROM de 8 koctets. Dans ce cas, seul 5 bits de l'octet contenant les poids forts du compteur ordinal seront reliés à la ROM externe.

Nous supposerons que la ROM est disposée en 8 blocs de 1 Koctet chacun.

Etant donné que les ROMs sont des produits du commerce, on considère leurs structures internes comme inconnues et l'apparition de pannes multiples comme probable. L'utilisation d'un code de contrôle double-rail est donc la solution la plus adaptée. Les contrôles s'effectuent de manière interne car chaque adresse et chaque donnée transitent obligatoirement par le bus interne qui est relié à un contrôleur double-rail. Cette structure sera décrite lors de la présentation du test en-ligne.

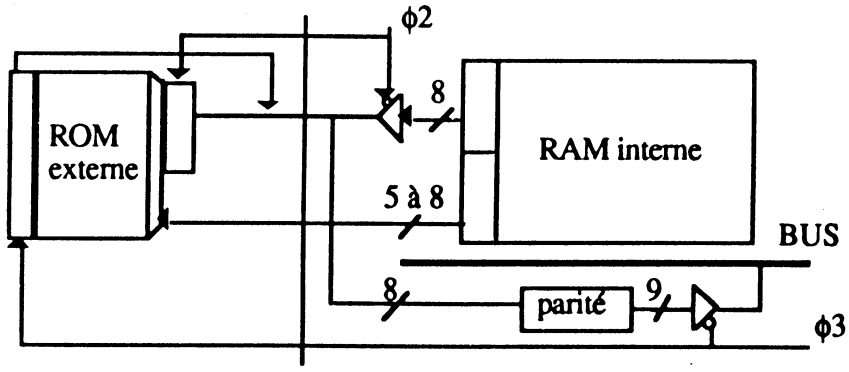


Schéma n°2 : Principe de communication avec la ROM externe

Req : L'utilisation de ROM externes de 9 bits permettrait d'éviter d'intégrer un générateur de parité mais demanderait une sortie supplémentaire. Une telle solution augmenterait sensiblement la sécurité par rapport aux pannes multiples de la ROM.

V.3 SPECIFICATIONS INTERNES

L'étude menée conjointement avec la CSEE, et à partir des descriptions présentées au chapitre n° 1, a permis de recenser les opérations que devrait effectuer le circuit et ainsi établir un jeu d'instructions. Le jeu d'instructions a volontairement été réduit au strict minimum , mais chaque instruction est utilisée très régulièrement contrairement aux circuits du commerce. Ceci est nécessaire pour une application de sécurité car tout bloc inactivé a beaucoup plus de chance d'être à l'origine de panne latente.

La fréquence de fonctionnement est de 1 MHz (Les applications envisagées sont relativement simples à traiter et ne nécessitent pas une grande vitesse de traitement)

La largeur des mots est de 8 bits (Le circuit dispose de 8 entrées et 8 sorties et toutes les instructions, sauf les branchements, sont codées sur 8 bits)

Le jeu d'instructions

CODE	nombre de cycle	DENOMINATION	
100aaaaa	4	RAM <- A	chargement
101aaaaa	4	A <- RAM	rangement
110aaaaa	4	RAM <- RAM + A + C	addition
111aaaaa	4	RAM <- RAM - A - C	soustraction
00100xxx	11	CO <- ROM	branchement
00101xxx	11	CO <- ROM si C	branchement si C
00110xxx	11	CO<- ROM si non C	branchement si non C
00111xxx	6	A <- ROM	valeur immédiate
01000xxx	4	C <- C ou C'	
01001xxx	4	C <- C et C'	
01010xxx	4	C <- NON C	
01011xxx	4	C <- 1	SET
01100xxx	4	C <- 0	RESET
01101a'a'a'	4	A -> C -> C'	PUSH
01110a'a'a'	4	A <- C <- C'	PULL
00001xxx	n+ 3	STOP = emission, reception et test	
00010xxx	i+ 3	PASSIVER	

Schéma n°2 : Jeu d'instructions

Pour effectuer ces instructions, l'utilisateur doit avoir à sa disposition :
Pour les opérations arithmétiques : (chargement, rangement, addition, soustraction, valeur immédiate)
- Un accumulateur A
- Une mémoire RAM de 32 mots
- Un registre de report C

Pour les opérations logiques : (push, pull et, ou, non, set, reset)
- 2 registres 1 bit C et C'
- L'accumulateur A

Pour les opérations de branchement : (branchement,branchement si C,branchement si nonC)
- aucun matériel supplémentaire

Pour les opérations système :
- L'instruction passiver permet de bloquer le circuit dans un état sûr
- L'instruction STOP permet de rafraîchir les données (émission réception) et d'effectuer le test hors-ligne périodiquement suivant un cycle contrôlé de 100 ms, c'est à dire que toute interprétation ne

débute que si le compteur est à 0. Cette instruction doit être présente dans tous les programmes, de telle sorte que l'on soit sûr d'avoir un test complet toutes les 100 ms (voir schéma).

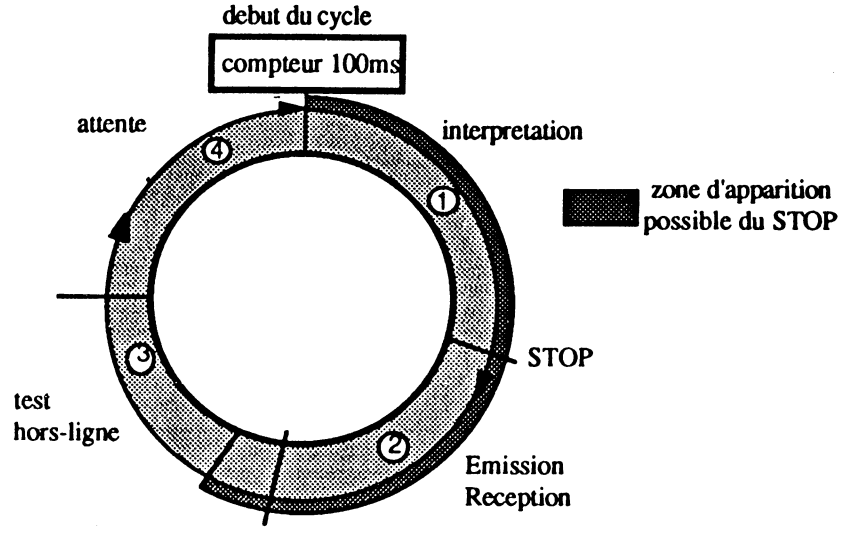


Schéma n°3 : Scrutation STOP

Cette instruction complexe est en fait un microprogramme établi à partir des instructions décrites précédemment et des ressources propres du MAPS. Elle permet aussi d'effectuer des temporisations qui seront des multiples de 100 ms.

La réception des messages consiste à prendre l'octet d'information contenu dans le message de 10 octets et de recalculer son codage à partir de constantes contenues en ROM externe. Le code contient 8 octets de codage arithmétique et 1 octet contenant les poids faibles de la date.

Pour valider le message, le micro-programme vérifie que l'écart de date (entre les poids faibles de la date reçue et les poids faibles de la date contenue en RAM interne) n'est pas trop important et éventuellement réajuste cette date. Si l'écart de date est trop important, le micro-programme le signale au poste d'aiguillage et attend qu'une nouvelle date complète (32 bits) lui soit envoyée. Cette date est stockée dans 4 mots de la RAM interne, elle est incrémentée à chaque cycle de 100ms. Il vérifie aussi si le codage arithmétique recalculé (2 fois 4 octets) est bien égal au codage arithmétique reçu. Si l'égalité n'est pas vérifiée, le micro-programme le signale et se fige dans un état sûr. Ces opérations s'effectuent sur 3 messages, l'un contient l'information et les deux autres contiennent un éventuel comptage d'essieux (modulo 512) et une indication de sens.

Lors de l'émission, le micro-programme code et date le message avant de l'envoyer et le poste d'aiguillage se charge de la validation lorsqu'il le reçoit.

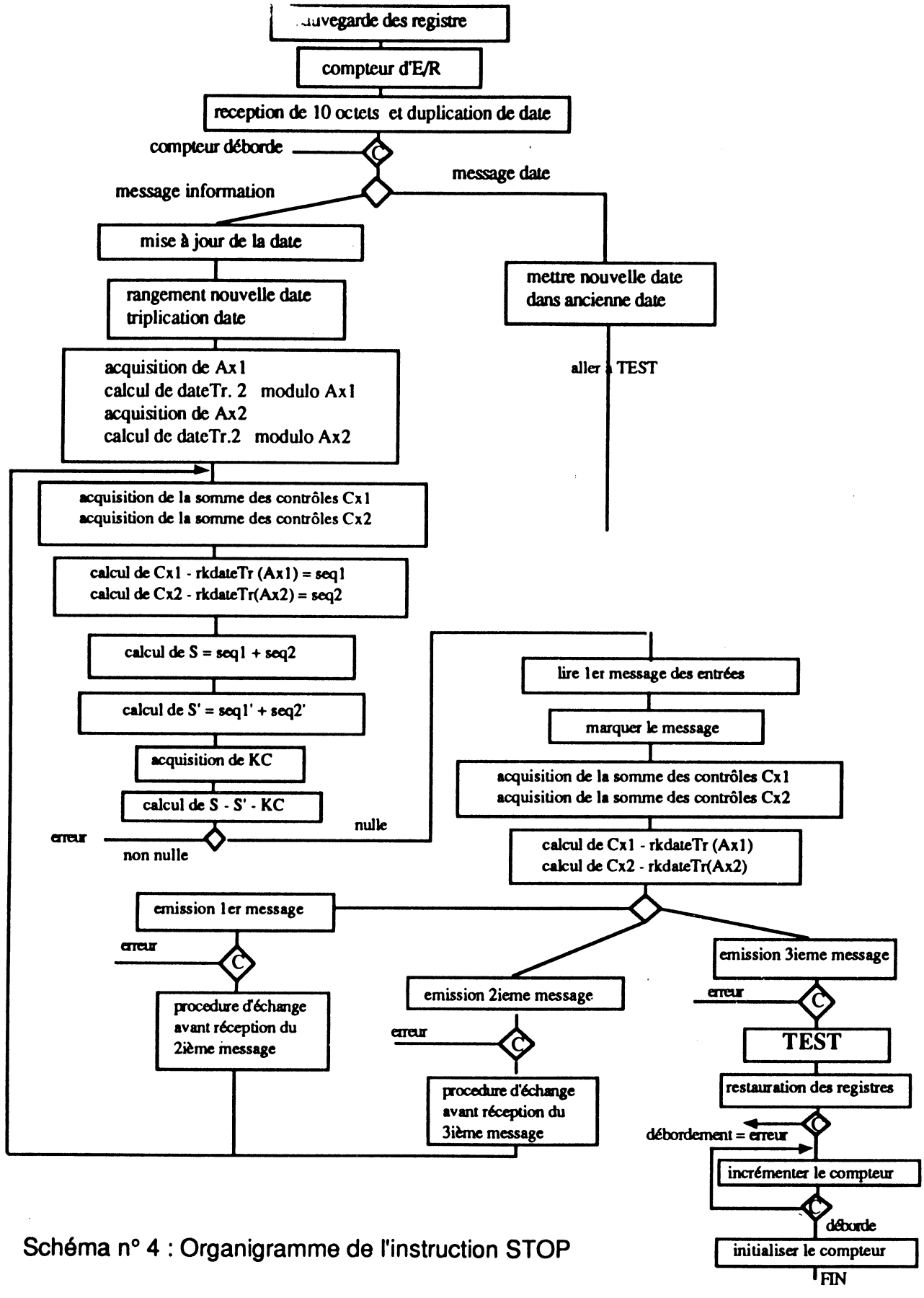


Schéma n° 4 : Organigramme de l'instruction STOP

V.4 ARCHITECTURE DU MICRO-CONTROLEUR MAPS

Le micro-contrôleur MAPS est un circuit du type micro-processeur, il dispose donc d'une partie contrôle de commande et d'une partie opérative d'exécution. La partie opérative est composée de tous les éléments qui viennent d'être déduits du jeu d'instructions (UA, Ul, RAM, registres...) et la partie contrôle gère le transfère des données entre blocs de façon à éviter les conflits d'accès à l'unique bus. Le micro-contrôleur MAPS dispose aussi d'un bloc d'entrée, d'un bloc de sortie et d'un bloc de communication. Cette architecture est décrite par le schéma n° 6. La suite de ce paragraphe va donc s'attacher à préciser la nature de chacun des blocs composant le circuit.

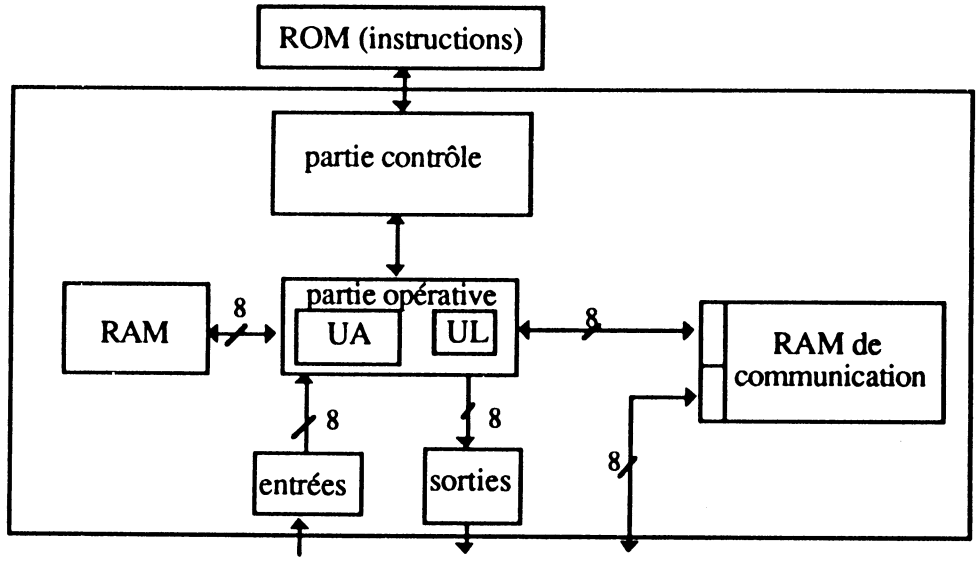


Schéma n° 6 : Architecture générale

V.4.1: La partie opérative

Le jeu d'instructions nous a permis d'avoir une première approche de l'organisation de la partie opérative. Outre les registres et les opérateurs mentionnés ci-dessus, le microcontrôleur a besoin d'autres ressources pour fonctionner correctement.

L'algorithme à exécuter se trouve dans une ROM externe, il faut donc nécessairement disposer d'un registre d'instructions (contenant l'instruction courante) et un compteur ordinal (contenant l'adresse de l'instruction courante). Le compteur ordinal a besoin d'être incrémenté (déroulement du programme) ou d'être chargé de valeurs particulières (branchement).

Cependant, afin de ne pas augmenter le nombre de blocs à tester, nous utiliserons l'unité arithmétique et deux registres de la RAM en guise d'incrémenteur.

Lors de l'instruction STOP, on exécute de nombreux calculs, il faut donc prévoir des registres supplémentaires pour ne pas perdre le contexte acquis. Pour cela on ajoute 32 mots à la RAM qui seront exclusivement réservés à la machine.
Pour éviter d'écraser le contenu du registre 1 bit C lorsque l'on exécute le programme externe on ajoute un registre 1 bit C" réservé principalement à l'incrémentation du compteur ordinal.
Le bus, volontairement choisi statique à cause de sa moindre sensibilité aux parasites externes, est la seule connexion directe qui nous permet de vérifier le bon fonctionnement de la partie opérative. Pour faciliter le test, il est préférable de n'utiliser qu'un seul bus par lequel passent toutes les données traitées par la partie contrôle .

Afin d'observer les sorties de l'unité arithmétique et de l'unité logique on les connecte au bus avant de les rediriger vers le registre choisi. La sortie de l'unité logique 1 bit est imposée sur les 8 bits du bus, ceci permet de la charger dans un bit quelconque de A sans laisser de ligne de bus flottante.
L'opération qui consiste à charger un des 8 bits de A dans C oblige l'UA à disposer d'un mode transparent.

On rajoute aussi un registre B en entrée de l'unité arithmétique pour effectuer des opérations d'addition et de soustraction. Cette solution a aussi l'avantage d'équilibrer les temps de propagation de l'UA. Le registre B n'est pas accessible à l'utilisateur.

L'utilisation d'une opération du type A <- A op B (couramment utilisée en micro-programmation) nous oblige à utiliser un registre A maître-esclave. De même, C est un registre maître-esclave car il doit encore fournir le report entrant lorsqu'il commence à recevoir le report sortant. C" qui a le même usage que C est donc aussi maître-esclave.
A partir de ces informations, il est maintenant possible de faire la synthèse complète de l'unité arithmétique et de l'unité logique.

V.4.1.1: Structure de l'unité arithmétique 8 bits

L'unité arithmétique est un additionneur qui effectue les opérations suivantes :
L'addition avec report, la soustraction avec report, le transfert de l'entrée 1 de A, le transfert de l'entrée 2 de B et l'incrémentation.

L'addition nécessite, en entrée un registre A et un registre B ainsi qu'un report entrant C ou C" et un report sortant C ou C".

La soustraction de nombres positifs utilise aussi l'additionneur mais le registre A d'entrée doit être complémenté ($B - A = B + \overline{A} + 1$ et $A - B - C = A + \overline{B} + \overline{C}$). Le report entrant C ou C" doit être complémenté, le report sortant C ou C" doit pouvoir être complémenté.
Il faut donc rajouter, devant l'entrée une "boîte" qui, soit laisse passer le contenu de A, soit le complémente, cette structure sera aussi imposée à B pour effectuer certains tests. De même, on

rajoute une boîte qui, soit laisse passer le report sortant, soit le complémente. L'inversion du report entrant sera effectuée au niveau de l'unité logique.

Pour effectuer le transfert de A ou de B, on utilise l'additionneur sous la forme A + 0 + 0 = A. Il faut donc rajouter en entrée de l'additionneur une "boite qui, soit laisse passer la valeur présente, soit la masque par la constante 00000000. Il faut aussi, dans ce cas, rajouter une boîte qui, soit laisse passer la valeur du report entrant, soit la masque par la constante 0.

Les opérations peuvent être effectuées sur plusieurs octets (4) mais il faut pour cela mémoriser le report sortant dans le registre 1 bit de l'unité logique.

V.4.1.2 : Structure de l'unité logique 1 bit

L'unité logique effectue les opérations ET, OU, NON, set, reset, C = A(ad'), C = report sortant, C = report sortant complémenté, C = C', C' = C, report entrant = C, report entrant complémenté = C complémenté.

Elle dispose donc à son entrée, soit d'un bit du bus par l'intermédiaire d'un multiplexeur 8 -> 1, soit du registre C', soit du report entrant ou du report entrant complémenté. Ces entrées sont sélectionnées par un multiplexeur 4 -> 1, la 4ième entrée sera une entrée test.
L'unité logique effectue l'opération C <- C op C' avec C = OU ou ET, la sortie passe par le bus et est transformée en 8 fois 1 bit.

On génère le complément de C en ajoutant à sa sortie une boîte qui, soit laisse passer C, soit son complément. Obtenir C complémenté en sortie demande que l'on effectue l'opération $\overline{C}$ OU 0, pour cela, il faut installer une boite à la sortie de C' qui, soit laisse passer C', soit le masque par la constante 0. En conséquence, la constante 0 peut être générée par l'opération C = ? ET 0.
Pour générer la constante 1, on effectue l'opération C = ? OU 1, le 1 est obtenu en ajoutant une boîte à la sortie de C, mais avant l'entrée de la boîte qui le complémente. Cette boite laisse passer le contenu de C, ou le masque par la constante 0 ($1 = \overline{0}$) .

Les opérations C' = 1 ou C' = 0 sont donc possibles en utilisant l'opération C' = C si l'on prélève la sortie de C après le passage par les deux boîtes.
De la même manière, la valeur de C' envoyée au multiplexeur 4 -> 1 est celle prélevée à l'entrée de la boîte de mise à zéro.

Le report est bien entendu prélevé après le passage par les deux boîtes pour pouvoir bénéficier, en entrée, soit de C, $\overline{C}$, 1 ou 0.

Toute les descriptions précédentes peuvent être résumées par le schéma fonctionnel n°7, lui même précisé par le schéma n° 8 au niveau du fonctionnement de l'unité logique et par le schéma n° 9 au niveau des registres d'entrée de l'unité arithmétique.

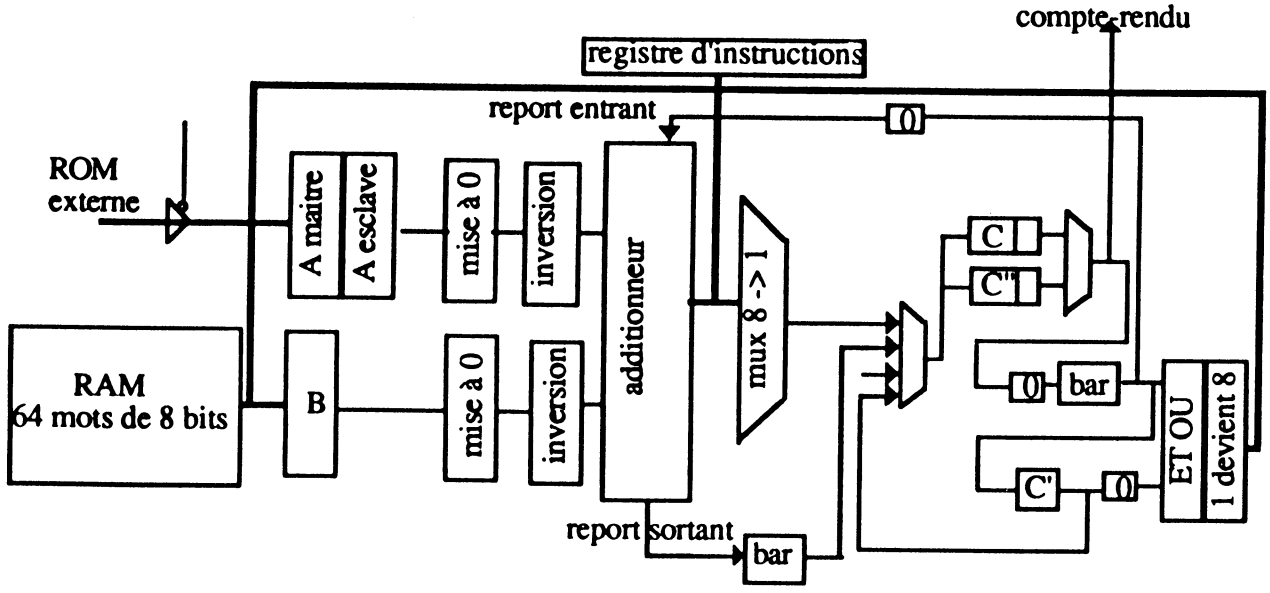


Schéma n°7 : Description fonctionnelle de la partie opérative

Les boîtes"bar" et "inversion" permettent d'inverser la sortie d'un registre sans en perdre son contenu. La boîte "1 devient 8" transforme le bit sortant de l'UL en 8 bits égaux.

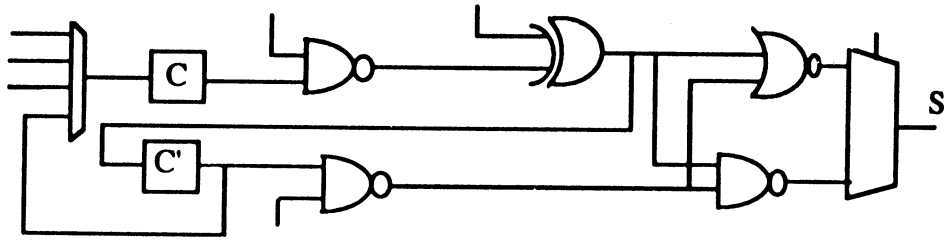


Schéma n°8 : Détail de l'unité logique

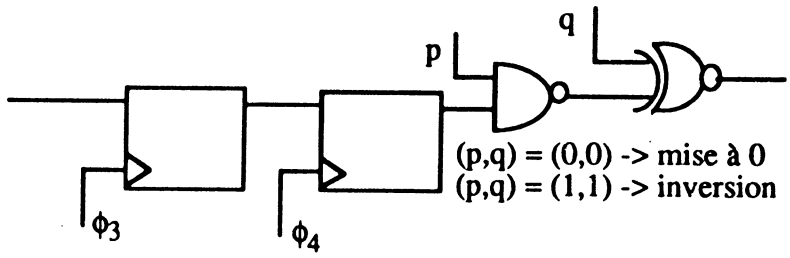


Schéma n°9 : détail d'une cellule d'entrée de l'additionneur

La partie opérative que l'on vient de présenter a dû être validée au niveau du transfert de registre. Cela a donc fait l'objet d'une simulation [COR 89] fonctionnelle et temporelle.

Le cycle machine d'une microseconde est décomposé en 4 phases comme le montre le diagramme du schéma n°5 .

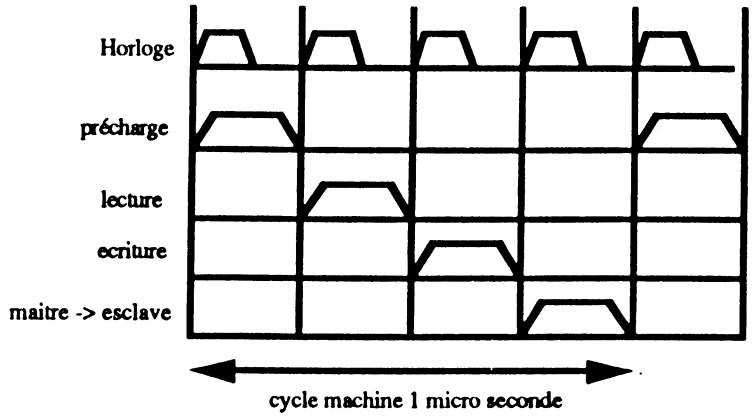


Schéma n°5 : Chronogramme

Sur chacune de ces phases s'effectuent des opérations bien déterminées.

- Sur $\phi 1$: Validation des signaux de la partie contrôle
  Précharge de la RAM

- Sur $\phi 2$: Lecture de la RAM
  Ecriture dans le registre B
  Ecriture dans la RAM double-accès

- Sur $\phi 3$ Ecriture de la RAM
  Ecriture des registres Amaitre, Cmaitre, C " maître, C ' et du registre d'instructions
  Mise sur le bus de l'UA, l'UL, des registres d'entrée et de sortie et de la ROM externe
  Lecture de la RAM double-accès

- Sur $\phi 4$: Ecriture dans les registres A esclave, C esclave et C" esclave
  Validation des comptes-rendus par la partie contrôle

V.4.2 : La partie contrôle

A partir de la description de la partie opérative on a pu déterminer la procédure qu'une partie contrôle devait exécuter pour faire fonctionner correctement cette partie opérative. L'organigramme présenté par le schéma n°10 donne le détail de ce fonctionnement. Notre choix architectural s'est orienté vers une structure classique très utilisée (68000) comprenant un PLA et une ROM de commande et de séquencement. Ce type d'architecture a d'ailleurs déjà fait l'objet d'une étude en vue d'obtenir une partie contrôle autotestable [NIC 90].

La partie contrôle agit en fonction du registre d'instructions, lui même chargé à partir du programme d'application contenu en ROM externe. Le déroulement du programme se fait par incrémentation du compteur ordinal ou par des branchements. Les commandes sont valides sur $\phi 1$ et maintenues sur les 4 phases. La nouvelle adresse générée en même temps que la commande ne peut être valide que sur $\phi 4$ car elle peut dépendre de la valeur du compte-rendu C (branchement).

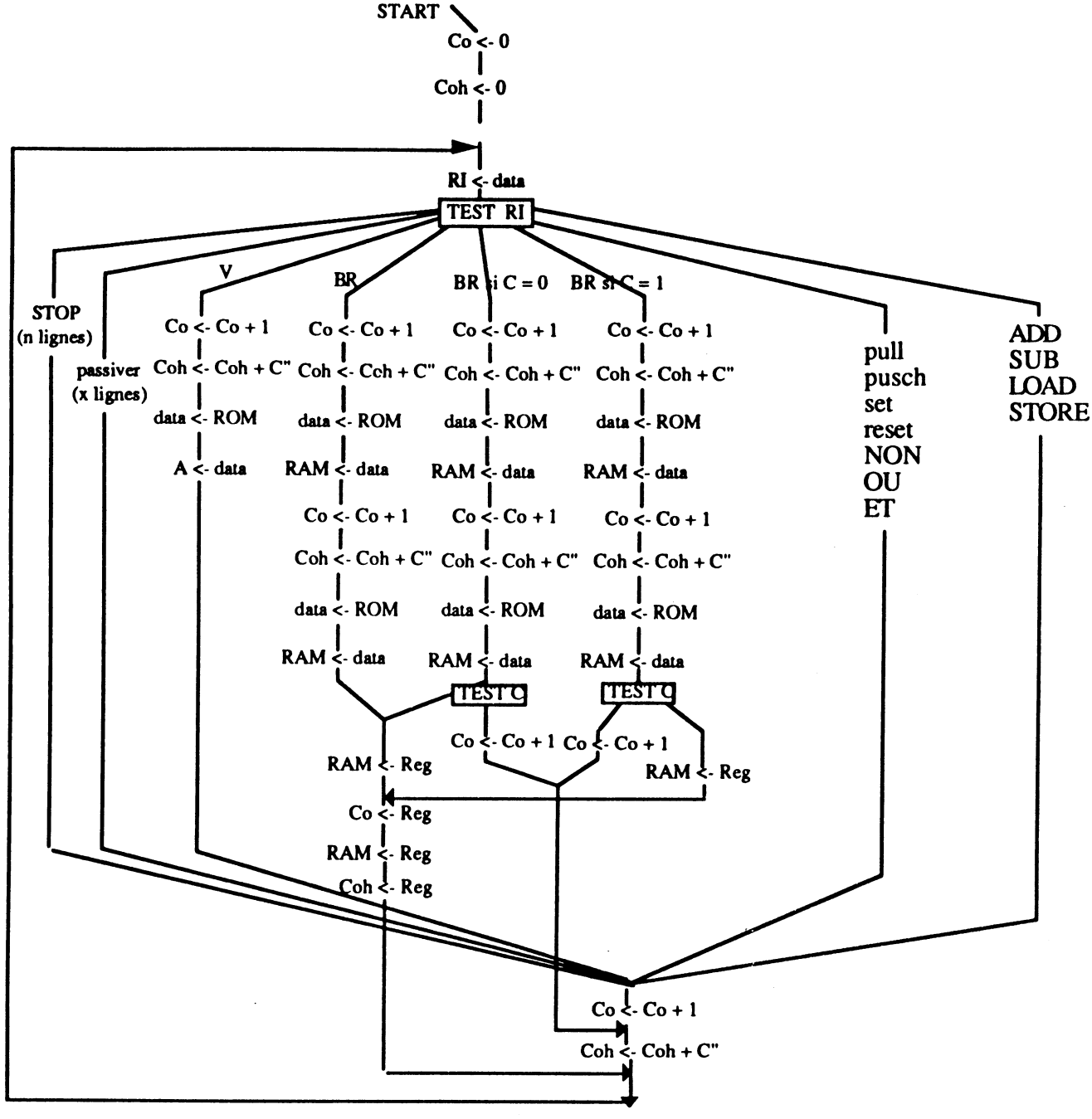


Schéma n°10 : Organigramme de fonctionnement de la partie contrôle

Dès que la micro-ROM est en présence de l'instruction STOP, elle se branche au début du programme d'exécution de cette instruction, puis elle effectue, de manière interne (sans lire RI) toute la procédure de transmission des messages, ainsi que le test hors-ligne. Cette procédure est décrite par l'organigramme du schéma n°4 .

Ce micro-programme demande de nombreuses instructions dont le taux de répétitivité peut être important (voir annexe). Cette constatation nous a permis de choisir une architecture constituée d'une micro-ROM et d'une nano-ROM directement adressée avec la micro-ROM. Cette architecture est utilisable car le circuit fonctionne avec une fréquence très faible, on peut donc diviser cette fréquence (voir schéma n° 11) de telle sorte la première ROM soit lue pendant que la deuxième est préchargée en $\phi'1$ puis que la deuxiéme soit lue en $\phi''1$. De plus, elle est beaucoup plus souple d'utilisation que la structure micro-ROM, nano-ROM du 68000 (voir schéma n° 11), qui demande de concevoir un décodeur spécifique pour adresser la nano-ROM en même temps, et avec la même adresse, que la micro-ROM. Cependant cette solution peut facilement être adaptée, elle ne nécessite pas d'aménagement de la phase f1 et est plus rapide.

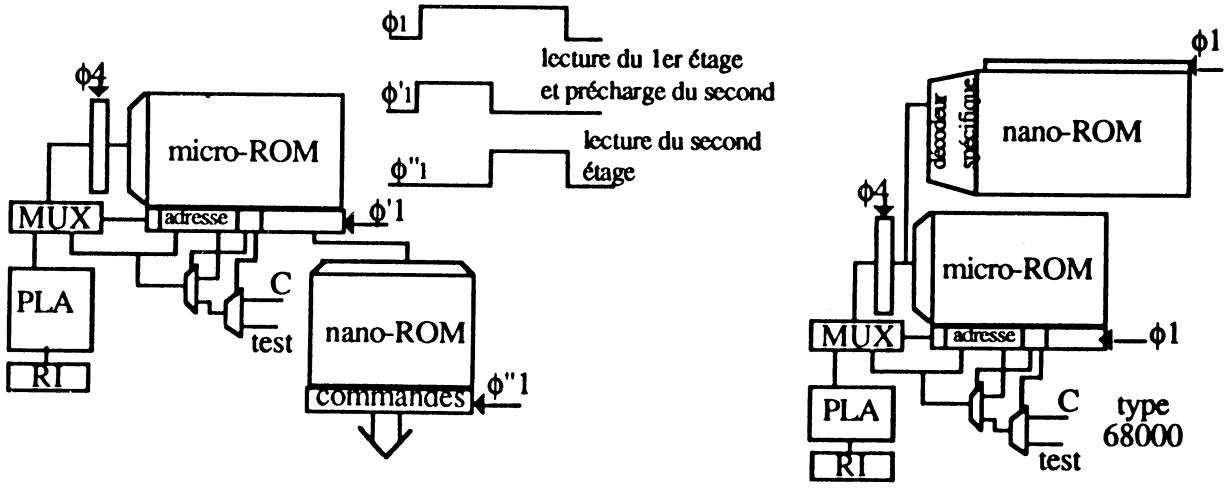


Schéma n°11 : Description de la partie contrôle

Le registre d'instructions (voir schéma n° 12) est un registre de 8 bits, dont 3 sont reliés exclusivement au PLA, 2 bits sont reliés à la fois au PLA et au registre d'adresses de la RAM, et les 3 derniers sont reliés à la fois au registre d'adresse de la RAM et à l'entrée d'un multiplexeur qui sélectionne un des 8 bits de l'accumulateur A pour le charger dans C.

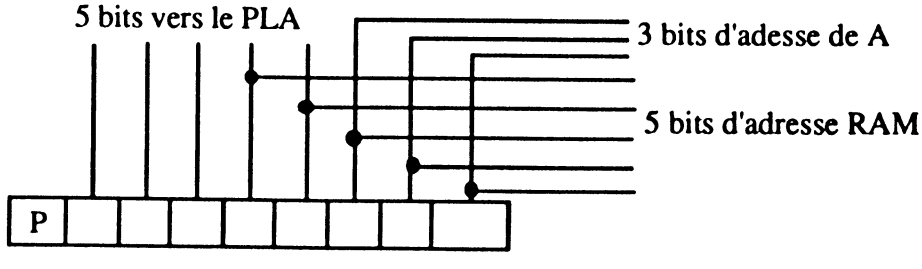


Schéma n° 12 : Organisation du registre d'instructions

V.4.3 : Les entrées

Les différents capteurs placés sur la voie ne procurent aucune protection aux données qu'ils envoient sur les entrées du circuit. Pour pallier à cet inconvénient, le circuit génère, pour chaque entrée, une sortie sur laquelle il envoie une séquence de 32 bits. Cette séquence est réintroduite sur l'entrée correspondante par l'intermédiaire d'un contact de sécurité conçu par la CSEE et non détaillé dans cette étude. Quand le contact est fermé, la séquence est envoyée sur l'entrée, et c'est l'état non sûr "1" qui est alors présent, par contre quand le contact est ouvert, la séquence ne peut pas être échangée et c'est l'état sûr "0" qui est présent.

La séquence générée par un LFSR, et la séquence réintroduite après son passage par la boucle externe, sont comparées. Si 31 des 32 bits sont égaux, alors l'entrée "1" peut être validée (voir schéma de principe n°13). Le choix d'utiliser 32 bits repose sur l'expérience acquise par la CSEE sur d'autres projets, mais comme le milieu extérieur est relativement perturbé il a semblé préférable de se donner 1 bit de tolérance.

La liaison entre l'entrée et le circuit se fait par l'intermédiaire d'un registre dont la lecture se fait sur $\phi 3$.

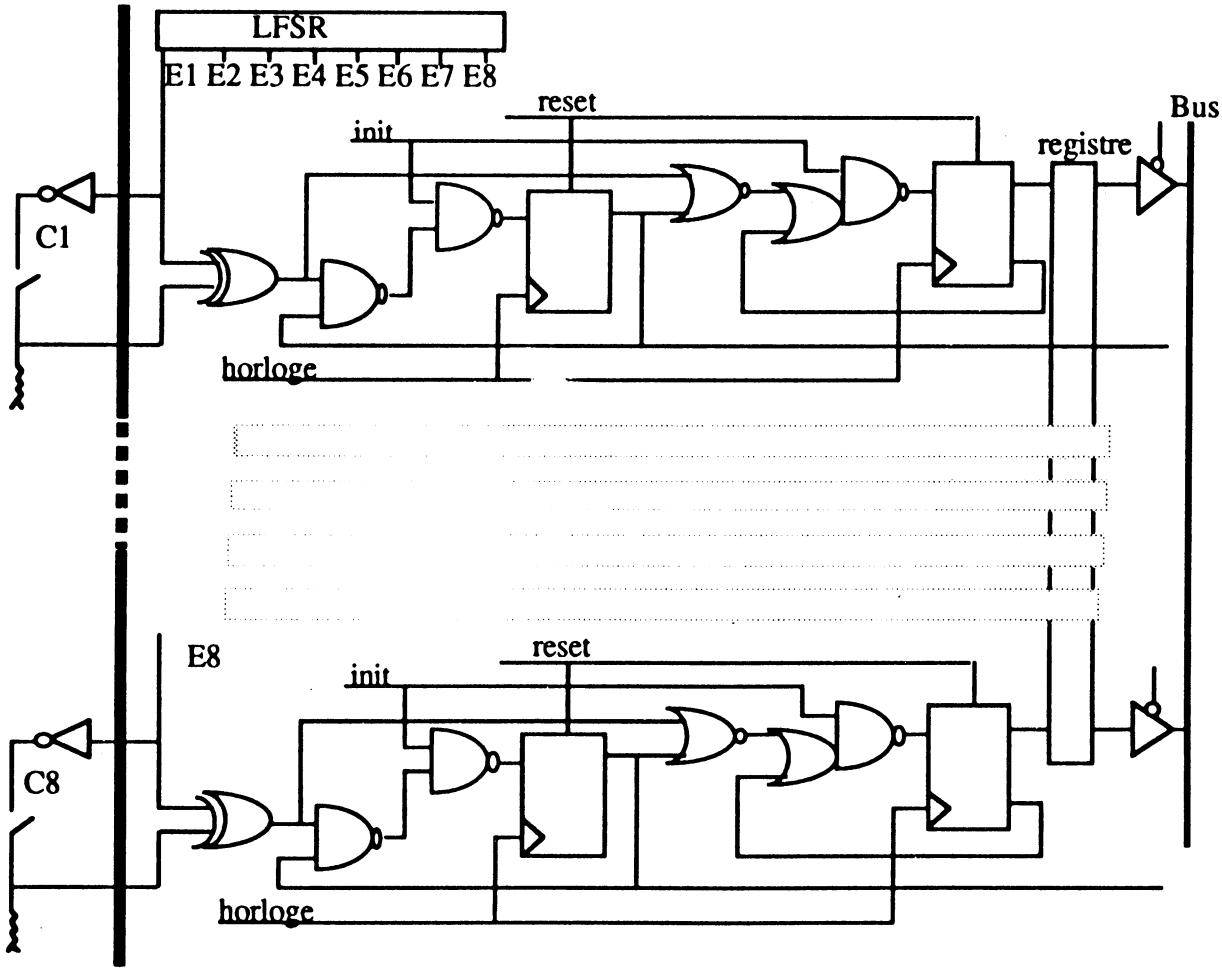


Schéma n° 13: Interface d'entrée

Deux de ces entrées peuvent avoir la fonction plus particulière de compter les essieux et de donner le sens des trains. Ces deux entrées sont reliées à un automate qui donne en sortie, sur deux nouveaux registres d'entrée, le nombre essieux (modulo 512) et le sens du train. Cette information est ensuite envoyée au poste d'aiguillage, qui, en fonction du sens, l'enverra au circuit directement en amont ou directement en aval. Ce second circuit effectue lui aussi un comptage d'essieux puis il le compare au comptage reçu, si les deux comptages sont égaux, la voie ou le canton de voie est libre.
Pour réaliser cette fonction particulière, les signaux provenant de la voie sont de deux types : ceux dits lents, dont le temps de permanence minimum à un état logique est supérieur à plusieurs dizaines de millisecondes, et ceux dits rapides dont les caractéristiques imposeront les limites de l'application.
Le schéma n°14 montre le timing des signaux rapides provenant des pédales. Les pédales sont des capteurs situés sur la voie ferrée qui indiquent le passage d'un essieu par l'émission d'une fréquence caractéristique.

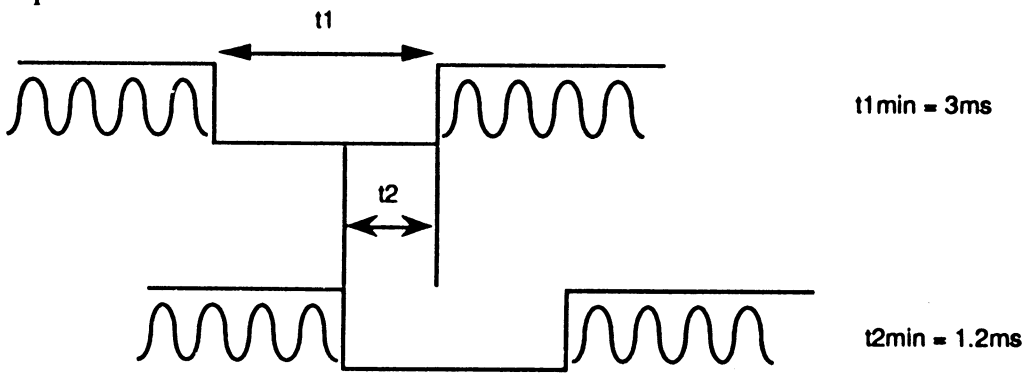


Schéma n°14 : Signaux provenant de la voie

A partir des données fournies par le timing ci-dessus nous avons pu déterminer les contraintes à respecter par le MAPS. En effet, pendant t2 min on doit pouvoir passer au moins deux séquences pseudo-aléatoires (tseq) ce qui nous donne:

$$tseq \leq (t2\ min) / 2 \text{ d'où : } tseq \leq 0.6ms.$$

La longueur de la séquence est de 32 bits. Cette valeur a été déterminée à partir d'études de sécurité faites antérieurement pour des systèmes similaires. La durée maximum dc chaque bit (tB) peut donc être déterminée :

$$tB \leq tseq /32 = 18.75\ \mu s$$

Ceci nous donne une fréquence d'horloge minimum de 53.3 kHz pour le générateur de séquences aléatoires. Notre choix s'est porté sur une fréquence de 62,5 KHz, soit (1 MHz/16). Il faut remarquer que les données provenant des entrées pédales sont traitées différemment de manière à

pouvoir effectuer le comptage des essieux et la détection du sens des trains. Le détail du fonctionnement sera exposé en même temps que le test.

V.4.4 : Les sorties

Les sorties du circuit sont du type mono-rail dynamisé en fréquence. Elles commandent chacun des actionneurs placés près de la voie avec une fréquence précise. L'une d'elles est directement contrôlée par la mémorisation d'erreur, elle indique donc à l'extérieur la présence d'une erreur interne en coupant l'alimentation du circuit. Ceci a pour effet immédiat de mettre le circuit dans un état sûr (fail-safe).

Tout autre état sur une sortie (signal électrique constant, autre fréquence ...) représente donc l'état logique sûr "0" . Les fréquences sont obtenues par division de la fréquence de base (1 MHz) à l'aide de compteurs et de décompteurs. Les différentes fréquences sont répertoriées dans le tableau n° 1, la déviation tolérée par le cahier des charges est de 2%.

Si	S1	S2	S3	S4	S5	S6	S7	S8	S9
fi (Hz)	1200	1600	2000	2800	4400	5200	6800	7600	9200

Tableau n°1 : Fréquences utilisées

Toutes ces fréquences sont des multiples premiers de 400Hz (fréquence très utilisée en signalisation ferroviaire) pour éviter tout problème lié à l'apparition d'harmonique. Pour des raisons de sécurité, les résultats des comptages et des décomptages sont ensuite comparés en permanence à l'aide d'un contrôleur double-rail comme le montre le schéma n°15.

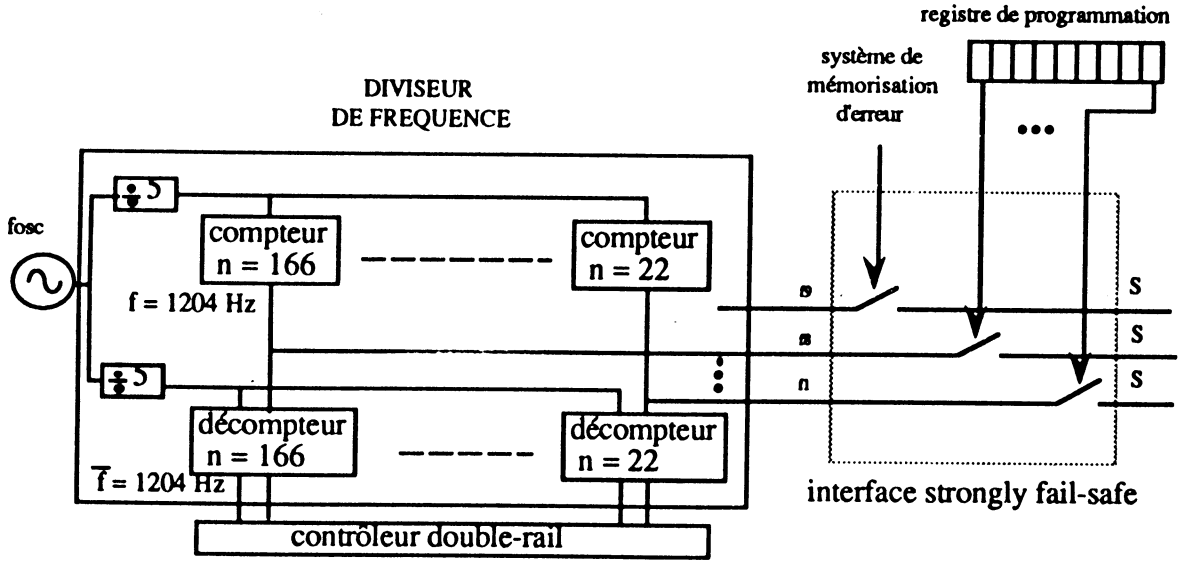


Schéma n°15: Organisation générale des sorties

V.4.5 : La communication

La communication avec le poste d'aiguillage est gérée par un microprocesseur externe qui traite tous les problèmes de protocole, son choix n'ayant pas été fixé, il ne fera l'objet d'aucun détail dans cette étude. Les entrées réservées à la communication ne bénéficient que d'une protection assurée par un codage arithmétique.

Ce microprocesseur est en permanence relié à une RAM interne double-accès qui sert donc d'intermédiaire lors des échanges de messages (émission/réception de l'instruction STOP).

Cette mémoire dispose de 62 octets : 30 pour la réception, 30 pour l'émission, 1 octet réservé au flag du message venant du MAPS et 1 octet réservé au flag venant du microprocesseur externe.

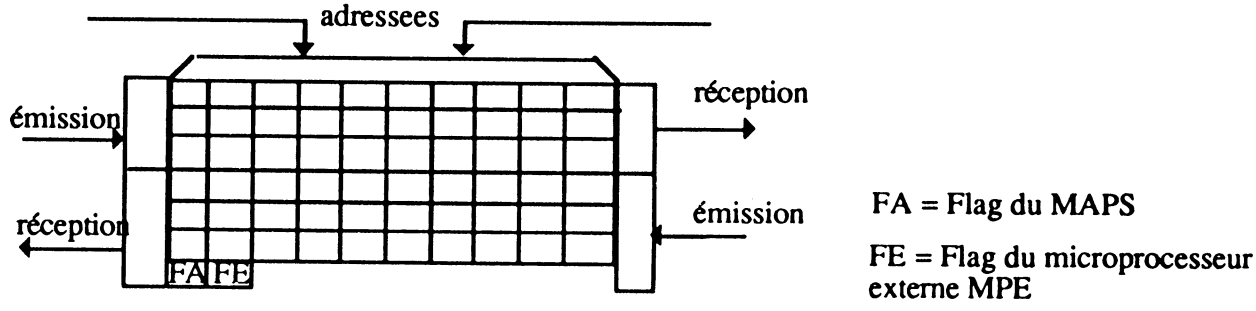
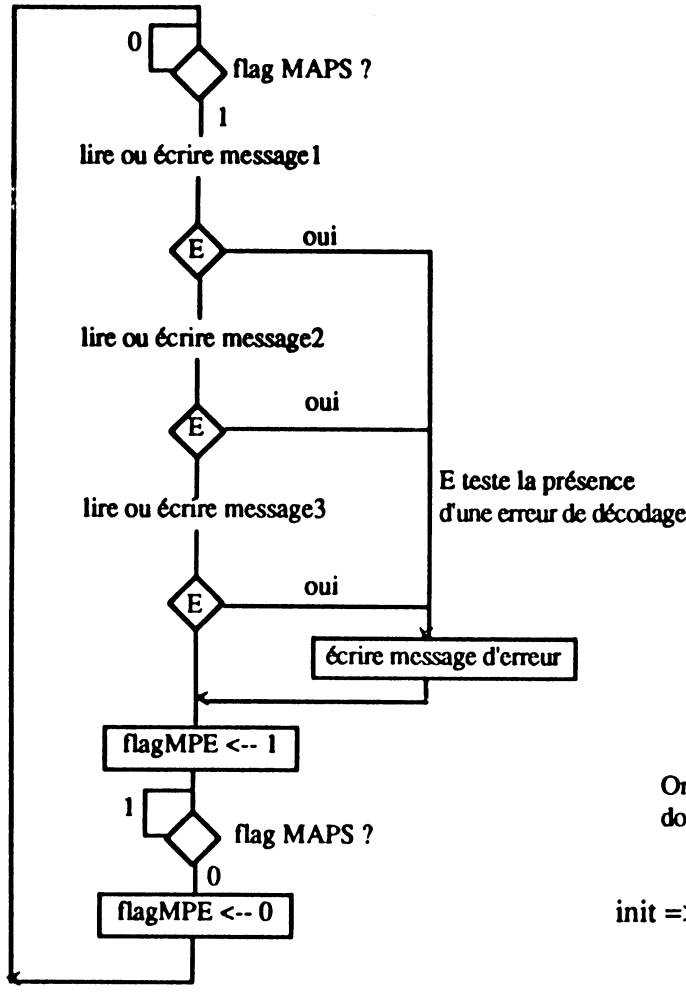


Schéma n°16 : Organisation de la RAM double-accès

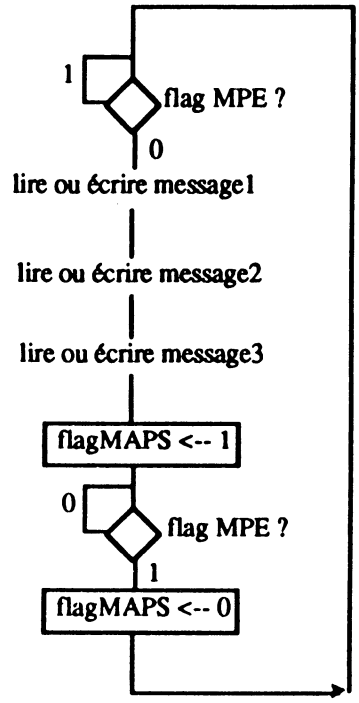
Lorsque le MAPS veut lire des informations dans la RAM double-accès, il génère l'adresse de son flag, contenu dans la RAM double-accès pour savoir s'il est autorisé à lire ou à écrire un message. Par exemple, si le bit de poids faible de l'octet du flag est à 1, il peut travailler avec la RAM, sinon il doit attendre en bouclant sur la procédure de lecture. L'accès à la RAM double-accès par le microprocesseur externe est basé sur le même principe. Ce fonctionnement est résumé par l'organigramme du schéma n°17 .

Il reste cependant un cas de conflit à résoudre. Ce conflit survient lorsque le MAPS veut lire l'octet contenant le flag de la RAM double-accès et que le microprocesseur est en train d'écrire (ou inversement) dans cette même case mémoire qui contient le flag.

Dans ce cas, soit le MAPS lit la bonne valeur, celle qui lui permet de commencer à dérouler son organigramme, soit la valeur lue l'empêche d'accèder à la RAM double-accès et il réitère sa lecture du flag (il reste dans la boucle d'interrogation du flag). Il ne lira donc la valeur du flag, qui lui permet d'accèder à la RAM double-accès, qu'après une procédure de lecture du flag supplémentaire. Ceci ne remet pas en cause le fonctionnement des communications.



Organigramme de communication entre la RAM double-accès et le MAPS



Organigramme de communication entre la RAM double-accès et le microprocesseur externe MPE

init => flagMAPS = flagMPE = 0

Schéma n°17 : Organigramme de communication

V.5 CONCLUSION

Ce chapitre nous a permis de mettre en place les éléments essentiels du circuit et de la carte, et de décrire le chronogramme permettant de les faire fonctionner de manière organisée. Toutefois, malgré notre volonté de traité le fonctionnel indépendamment de tout test, nous avons vu que certains choix architecturaux étaient directement liés au test. Le but du prochain chapitre aura donc pour objectif, à partir de l'architecture et des bases théoriques du test, d'obtenir un circuit capable de traiter des problèmes précis tout en assurant son propre test.

CHAPITRE VI

TEST EN-LIGNE DU CIRCUIT

VI.1 INTRODUCTION

Le but de ce chapitre est de présenter toutes les modifications que l'on a apportées à l'architecture de base décrite au chapitre précédent pour obtenir un circuit auto-contrôlable en-ligne capable de détecter les pannes triples. Pour cela, nous allons prendre chacune des grandes parties du circuit en faisant apparaître les modifications. De manière générale, pour le coeur du circuit, ces modifications sont basées sur l'utilisation du code double-rail et parité pour la partie opérative, et sur l'utilisation du code double-rail, parité et Berger, pour la partie contrôle. Ce choix nous a permis de présenter le fonctionnel (chapitre V) indépendemment du test car les intéractions sont beaucoup plus faibles.

VI.2 ARCHITECTURE GENERALE

Le test en-ligne du micro-contrôleur MAPS est assuré en utilisant le principe de la redondance hétérogène . Pour cela, la partie opérative et la partie contrôle sont dupliquées de manière duale et leurs résultats sont en permanence comparés à l'aide de contrôleurs double-rail. De plus, on associe à chacune des données de 8 bits son bit de parité pair. En conséquence, chaque donnée dispose de 8 bits d'information et de 1 bit de codage , soit un codage total disposant de 8 bits duaux + 2 bits de parité. L'utilisation du code de parité en complément du code double-rail permet d'augmenter la sécurité du système vis à vis des pannes multiples. Cette augmentation du nombre de bits du code a été entérinée par une étude de l'INRETS (Institut national de recherche sur les transports et leur sécurité). Cette étude montre que parmi deux solutions possibles: utilisation de 2 bits de parité pour coder les deux parties opératives, ou l'utilisation de 2 bits de parité pour coder les deux groupes de 4 bits d'une seule partie opérative, la première solution est beaucoup plus efficace contre la présence de parasites. En conséquence, les contrôleurs devront aussi contrôler la parité. L'architecture générale est décrite par le schéma n° 1. Les résultats de tous les contrôleurs double-rail intermédiaires sont recueillis par le contrôleur double-rail global.

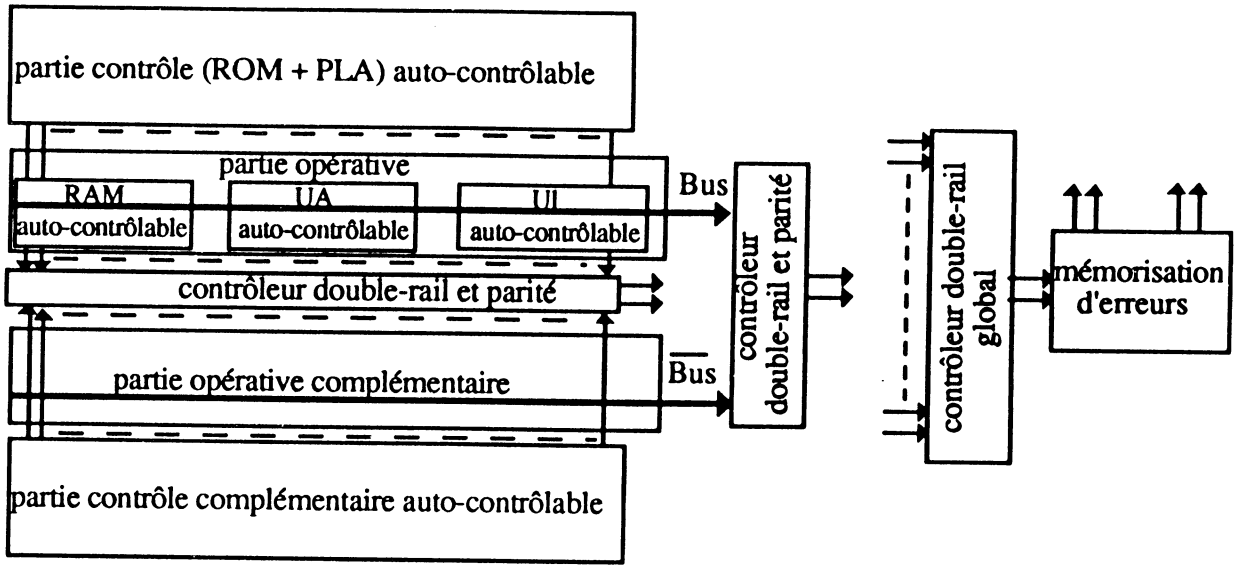


Schéma n° 1 : Aspect du circuit auto-contrôlable

Dans la suite de ce chapitre, on décrit successivement la partie opérative puis la partie contrôle en détaillant chacun des blocs qui les composent. En fait, nous ne décrirons que les blocs self-checking (RAM, UA...) d'une seule partie opérative et d'une seule partie contrôle. Les blocs de la partie opérative duale et de la partie contrôle duale sont, soit les mêmes, soit les duaux de ceux qui sont décrits. Cette architecture nous permet d'assurer la détection de toutes les pannes survenant dans un seul bloc ainsi que des pannes doubles et triples survenant dans tout le circuit.

VI.2.1 : La partie opérative

La démarche que l'on va suivre consiste à détailler chacun des blocs composant la partie opérative

VI.2.1 .1 : La RAM self-checking

Le code détecteur d'erreur utilisé est un code double-rail, la conception du plan mémoire peut être quelconque, du moment que l'on respecte les règles d'alimentation et d'éloignement des deux blocs duaux. On peut cependant remarquer qu'il est assez aisé de dessiner un plan mémoire contrôlé avec un code de parité puisque, de manière générale, la règle R1 est vérifiée. La règle R' 2 peut être vérifiée en appliquant la conception décrite par le schéma n°2 .

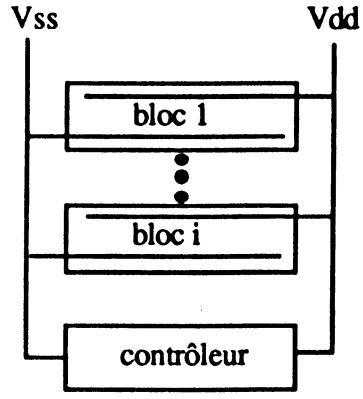


Schéma n°2 : Exemple d'application de la règle R'2

La règle R3 est vérifiée car les cellules voisines appartiennent à des mots différents (multiplexage colonne).
La règle R4 est vérifiée puisque que l'on peut implanter les lignes Vss et Vdd alternativement.
Toutefois, certains défauts du décodeur ligne et du décodeur colonne peuvent introduire des erreurs multiples à la sortie de la RAM. Pour détecter ces pannes on contrôle les décodeurs avec le mécanisme présenté par le schéma n°3.

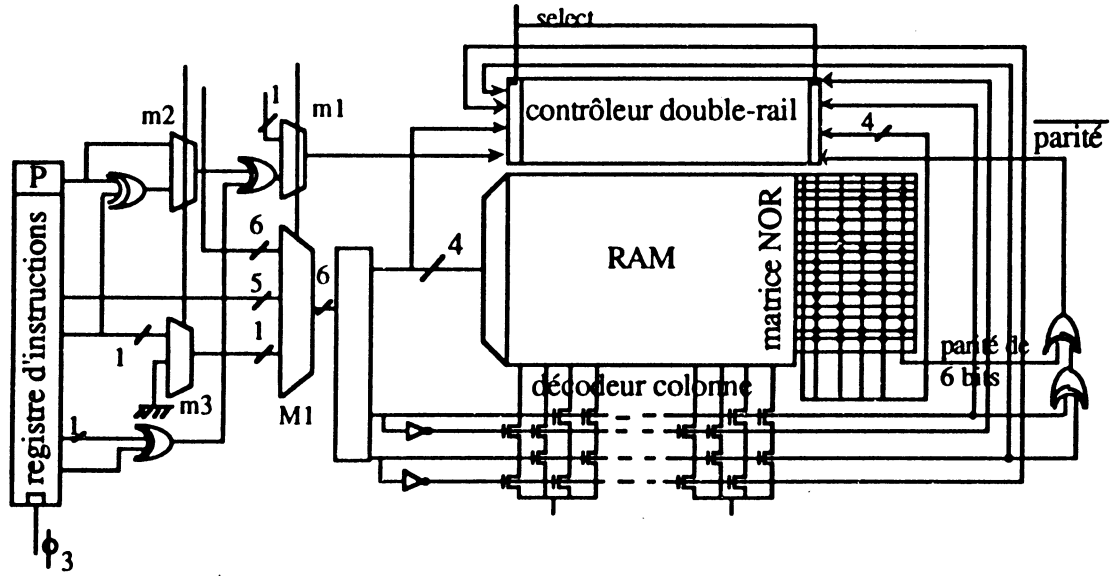


Schéma n°3 : RAM "self-checking"

La technique utilisée pour le décodeur ligne consiste à régénérer les adresses complémentaires après traversée du plan mémoire et à les comparer avec les adresses originelles. Nous obtenons ainsi un décodeur SFS (strongly fault secure) pour toute panne simple et multiple [NIC 90b] Le contrôle

des adresses colonne peut se faire directement, à l'aide d'un contrôleur double-rail placé à la sortie du décodeur colonne ou bien, comme le montre le schéma n°3 en utilisant un seul contrôleur double-rail.

On remarquera que la matrice NOR régénère aussi la parité des 6 bits de l'adresse. Cependant, la parité donnée par le registre d'instructions est la parité des 8 bits du registre. Il faut donc s'assurer que l'on compare bien les mêmes parités dans tous les cas de fonctionnement.

- 1er cas : fonctionnement normal

Le multiplexeur m2 laisse passer P⊕ bit6, le multiplexeur m3 impose 0 et la parité envoyée au contrôleur double-rail est P ⊕ bit6 ⊕ (bit7⊕bit8) = P ⊕ 0 ⊕ (bit7⊕bit8).

- 2ième cas : test de la RAM

Le multiplexeur m2 laisse passer P, le multiplexeur m3 laisse passer la valeur du bit6 et la parité envoyée au contrôleur double-rail est P ⊕ (bit7⊕bit8) .

- 3ième cas : fonctionnement interne avec la micro-ROM

Le multiplexeur m1 laisse passer la parité du champ d'adresses provenant de la micro-ROM et le multiplexeur M1 laisse passer l'adresse provenant de la micro-ROM.

Le rattrapage du retard occasionné par la régénération des adresses sur deux cycles différents nous oblige à implémenter un registre devant les entrées du contrôleur double-rail. Ce registre est commandé par le signal select de la RAM, c'est à dire que le contrôle des erreurs se fait sur $\phi 2$ si c'est une instruction de lecture qui est demandée et sur $\phi 3$ si c'est une écriture.

VI.2.1.2 : L'unité arithmétique

Toutes les opérations sont effectuées sur 8 bits, c'est à dire que l'opération n'est contrôlée que par le code double-rail et non plus par le code double-rail et parité. En conséquence, pour rétablir le même niveau de contrôle dans l'additionneur on implante des additionneurs "self-checking" dans chacune des deux parties opératives. Leur architecture générale est décrite par le schéma n°4 .

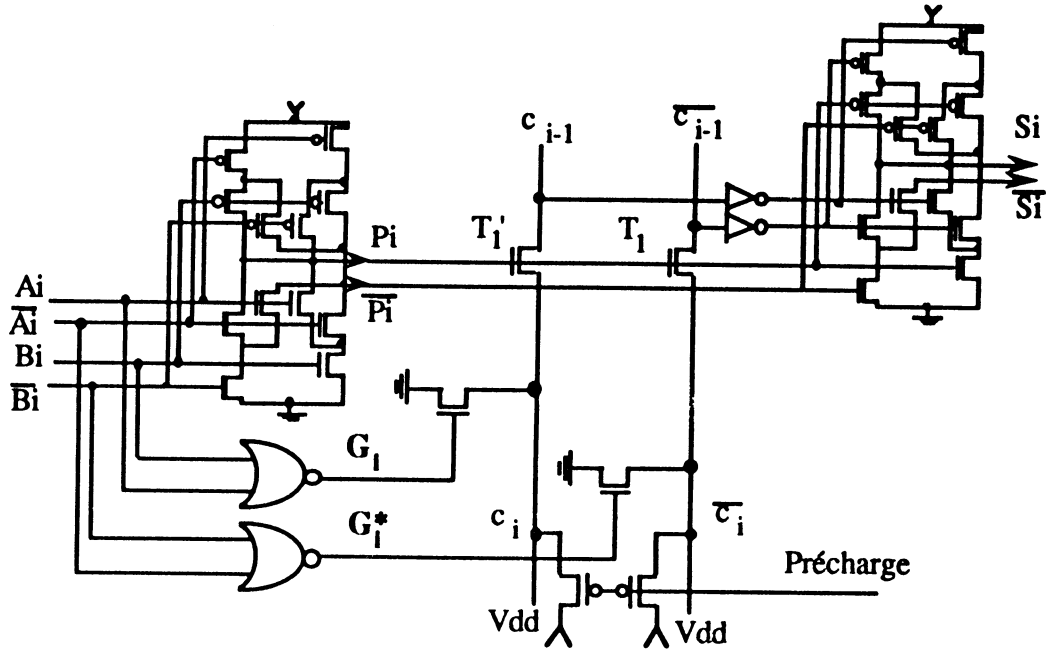


Schéma n° 4 : Unité arithmétique "self-checking"

Cette unité arithmétique a été présenté dans [NIC 86] . Elle fournit des sorties duales qui forment un code non-ordonné, elle respecte les règles R5 à R8 et assure la propriété "strongly fault secure". Cette unité arithmétique est beaucoup plus compacte que l'utilisation de deux unités arithmétiques duales . Le fonctionnement des deux unités arithmétiques peut être décrit par le schéma n°5.

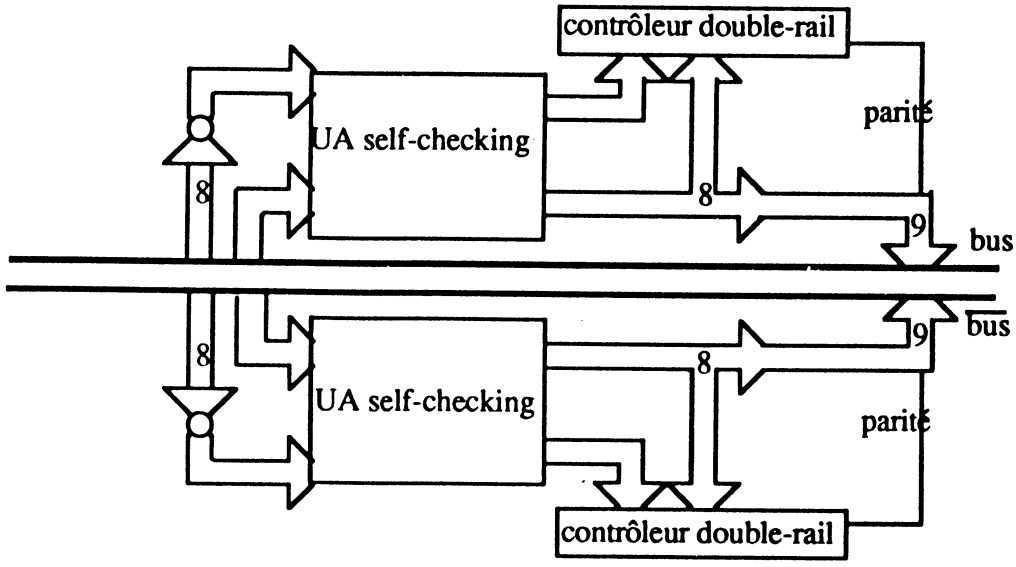


Schéma n°5 : Architecture des unités arithmétiques

VI.2.1.3 : L'unité logique

Le principe de sécurité est le même que celui utilisé pour l'unité arithmétique, sauf qu'ici, les unités logiques "self-checking" sont conçues avec deux unités logiques duales . Il n'y a aucun problème topologique pour placer ces deux unités logiques 1 bit puisqu'elles sont incluses à l'intérieur d'une structure bit slice de 9 bits (voir schéma n°6). Les sorties de l'unité logique sont transformées en 8 sorties égales (à 1 ou à 0) auxquelles on ajoute une parité toujours nulle pour être contrôlées par le contrôleur du bus.

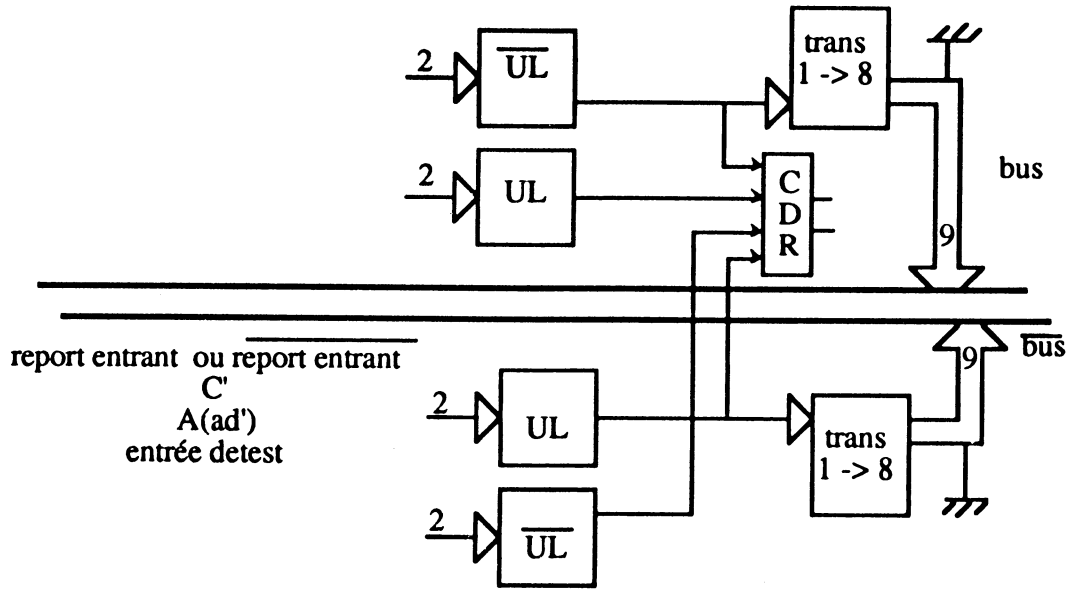


Schéma n°6 : Architecture des unités logiques

VI.2.2 : La partie contrôle

La partie contrôle est composée de deux parties duales, chacune d'entre elles étant "self-checking". elles reçoivent leurs informations par l'intermédiaire de registres d'instructions. Le PLA décode le code opération et génère l'adresse de la micro-ROM. Cette adresse peut aussi provenir du champ "micro-adresse suivante" de la micro-ROM. Le choix entre les deux sources est déterminé par un multiplexeur commandé par le signal T, généré par la micro-ROM. La micro-ROM assure le séquencement des instructions tandis que la nano-ROM reçoit les sorties de la micro-ROM et génère les commandes de la partie opérative.

VI.2.2.1 : Le PLA

Les sorties du PLA sont divisées en deux champs. Le premier champ est destiné au décodeur ligne de la micro-ROM, il est codé avec le code de Berger. Le second champ est destiné au décodeur colonne, il est codé avec le code double-rail. Ces codes étant des codes non-ordonnés, ils

assurent la propriété SFS (strongly fault secure) pour le PLA [MAK 82] cette propriété est assurée de la même façon pour les PLAs implémentés en logique CMOS préchargée (voir annexe V). Le contrôle est effectué au niveau de la micro-ROM sans qu'il y ait besoin de rajouter un contrôleur.

VI.2.2.2 : La micro-ROM "self-checking"

La micro-ROM et la nano-ROM ont les mêmes structures et seront traitées de la même façon pour les transformer en ROM self-checking.

Un champ de la micro-ROM (schéma n° 7) régénère les compléments des entrées du décodeur ligne (code de Berger). Les entrées du décodeur ligne et ses compléments générés par la micro-ROM sont comparés par un contrôleur double-rail, ainsi, on assure la propriété SFS pour le décodeur ligne et ceci pour toute panne (simple, multiple etc) [NIC 90b]. Les entrées du décodeur colonne (multiplexeur) étant codées dans le code double-rail, sont contrôlées par un contrôleur double-rail, directement après avoir traversé le multiplexeur.

Il ne nous reste plus qu'à assurer la propriété SFS pour le plan mémoire de la micro-ROM et pour le multiplexeur. On peut vérifier que les plans mémoire respectent les règles R1 et R2 (chaque ligne est connectée avec une seule sortie [NIC 85]). En conséquence,on peut coder le contenu de la micro-ROM avec n'importe quel code détectant les erreurs simples.

Ainsi, les sorties duales des deux nano-ROMs, qui représentent les champs de commandes sont, comme la partie opérative, contrôlées par des contrôleurs double-rail et parité.

Ces contrôleurs auront une plus grande efficacité s'ils se trouvent en aval des deux parties opératives.

En conséquence, pour obtenir des micro-ROMs et des nano-ROMs self-checking (SFS) pour tout type de panne (simple, multiple et à caractère séquentiel) il suffit de concevoir les ROMs suivant le principe du schéma n°7.

La compensation des retards dus à la régénération des adresses sur deux phases différentes est assurée grâce à des registres.

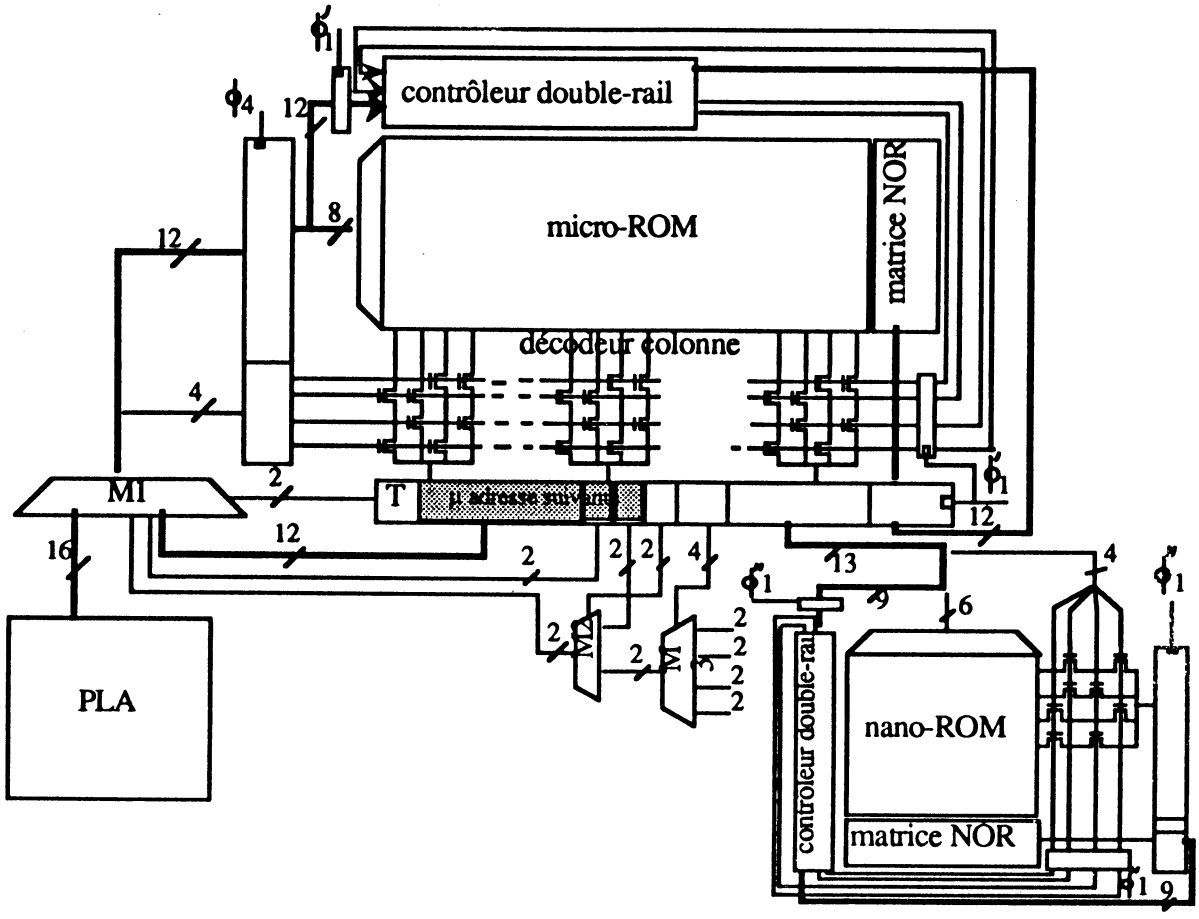


Schéma n°7 : Partie contrôle self-checking

La micro-ROM génère aussi la commande du multiplexeur M1 codée en double-rail pour sélectionner soit l'adresse venant du PLA, soit l'adresse suivante codée, générée par la micro-ROM. Elle génère aussi, en double-rail, le bit de commande du multiplexeur M2 qui sélectionne soit le compte-rendu C (et son dual), soit le bit de poids faible (codé en double-rail) de la micro-adresse suivante (ou sélection des colonnes). Elle génère également, codé en 1 parmi 4 les 4 bits de commande du multiplexeur M3 qui selectionne le compte-rendu approprié (C, test, compteur de cycle) et son dual.

Le choix d'utiliser un code 1 parmi 4 découle d'un choix du fonctionnement de la micro-ROM. lors du remplissage de la micro-ROM, chaque fois que l'on est en présence d'un branchement, on choisit l'adresse paire immédiatement suivante comme "adresse suivante".

adresse colonne paire immédiatement suivante avec son code associé	test	nouvelle adresse
00 -> 0101	C = 0	0101
00 -> 0101	C = 1	0110
10 -> 1001	C = 0	1001
10 -> 1001	C = 1	1010

Tableau n° 1 : Fonctionnement du bit de remplacement

Le remplacement des bits se fait donc sur les deux derniers bits de l'adresse colonne codée en double-rail, C pour l'avant dernier et $\overline{C}$ pour le dernier. Les remplacements s'effectuent de la même façon si au lieu de C on sélectionne le compteur, ou un compte rendu de fin de test , chacun d'eux associés à son dual respectif.

Les multiplexeurs M1 et M2, du type 2 -> 1 sont implémentés en CMOS complémentaire avec une porte complexe inverseuse comme cela est présenté par le schéma n° 8.

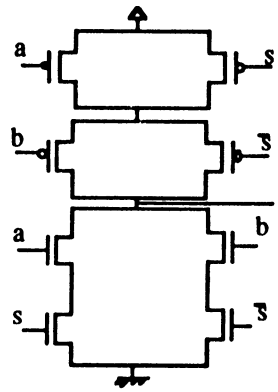


Schéma n°8 : Multiplexeur 2 --> 1

Les erreurs qui peuvent survenir sont du type 00 ou 11 sur les lignes de commande. Les différents cas possibles sont résumés dans le tableau n°2.

S	$\overline{S}$	a	b	sortie
0	0	0	0	1*
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
1	1	0	0	1*
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0*

Tableau n°2 : Récapitulations des erreurs possibles

Les * indiquent les valeurs qui restent correctes malgré les erreurs des lignes de commande. Le multiplexeur M2 reçoit en entrée, soit les bits de poids faibles de l'adresse paire immédiatement suivante (0,1), soit les bits duaux du compte-rendu (1,0) ou (0,1). La combinaison (0,1) en remplacement de (0,1) en présence de l'erreur $(S,\overline{S}) = (0,0)$ donne le résultat (1,1) en sortie qui est hors code, par contre si l'erreur est $(S,\overline{S}) = (1,1)$ le résultat devient (0,1) qui est le résultat correct inverse de l'entrée. La combinaison (1,0) en remplacement de (0,1) en présence de l'erreur $(S,\overline{S}) = (0,0)$ donne le résultat en (1,1) en sortie qui est hors code et si l'erreur est $(S,\overline{S}) = (1,1)$ le résultat devient (0,0) qui est aussi hors-code. Les mots hors-code transitent par le multiplexeur M1 puis par le décodeur de la micro-ROM. Ces mots hors-code commandent le décodeur colonne et sont directement reliés au contrôleur double-rail, l'erreur est donc détectée.

Pour le multiplexeur M1, le raisonnement se fait sur 2 fois 16 bits globalement différents l'un de l'autre, sinon le microprocesseur se trouve dans une boucle infinie.
Si, seule l'adresse colonne est différente, on se retrouve dans le même cas que celui du multiplexeur M2.
Si l'erreur est $(S,\overline{S}) = (0,0)$, alors toutes les sorties inverses sont à 1, le mot est forcément hors-code et le contrôleur double-rail qui contrôle le décodeur de la ROM détectera l'erreur. Si l'erreur est $(S,\overline{S})$ = (1,1) chaque fois que les adresses sont différentes, la sortie inverse se trouve à 0 dans l'adresse et dans le code c'est à dire le non ou-exclusif. Pour que le circuit transmette au décodeur une mauvaise adresse, mais qui appartiendrait au code, il faudrait que si $a \oplus b = \overline{c}$ alors (nbr de 0 de a)$\oplus$(nbr de 0 de b) = (nbr de 0 de c) ce qui est impossible.

Le multiplexeur M3 est du type 4 -> 1, il est donc intéresssant de coder ses commandes en 1 parmi 4 au lieu d'utiliser le code double-rail, si l'on peut prouver comme pour M1 et M2 qu'il n'est

pas nécessaire de tester les commandes, car la porte complexe de base demande moins de transistors. Cette porte est présentée par le schéma n°9 .

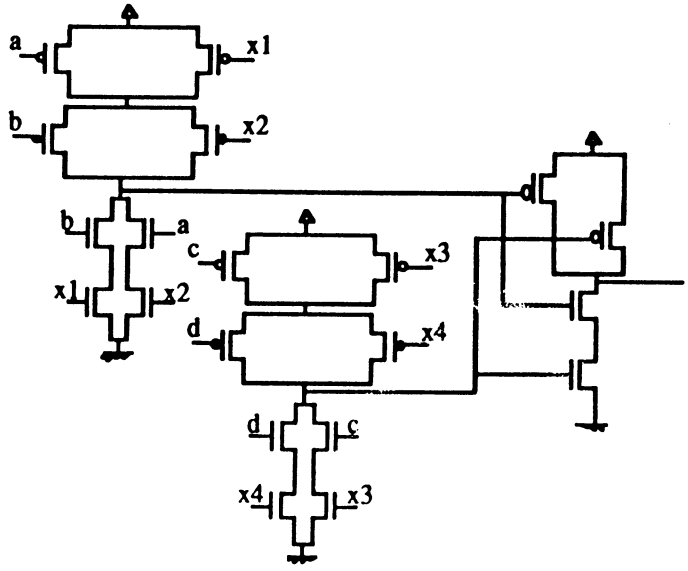


Schéma n°9 : Multiplexeur 4 --> 1 utilisé avec un code 1 parmi 4 en commande

Si l'erreur est du type 0000, les sorties inverses des deux multiplexeurs seront à 1, cette erreur sera ensuite propagée jusqu'au décodeur colonne et l'erreur sera détectée par le contrôleur double-rail. Si l'erreur est du type 0011, ou tout autre erreur possédant deux 1, les sorties sont correctes si les deux valeurs sélectionnables sont égales, sinon les sorties sont hors-code et le décodeur colonne le détectera.

Ceci nous permet donc d'affirmer qu'il n'est pas nécessaire de contrôler les commandes des multiplexeurs à la sortie des multiplexeurs.

Le code de la micro-commande qui est envoyé à l'entrée de la nano-ROM est codé à l'aide du code de Berger (6 + 3 bits) et du code double-rail (2 + 2 bits). La nano-ROM est contrôlée selon le principe de la régénération présenté ci-dessus. Les sorties de la nano-ROM attaquent directement la partie opérative.

La capacité des ROMs est au maximum de 1024 destinations adressables pour la micro-ROM et de 256 destinations adressables pour la nano-ROM.

Le contrôleur double-rail de la micro-ROM a donc 16 paires d'entrées et celui de la nano-ROM a 13 paires d'entrées. Leur conception comme celle de tous les contrôleurs du circuit est décrite par le schéma n°10 .

VI.3 LES CONTROLEURS DOUBLE-RAIL

Les contrôleurs sont tous conçus sur le même modèle.

Pour la partie contrôle (micro-ROM, nano-ROM, PLA, multiplexeurs) nous avons rajouté deux contrôleurs double-rail, l'un pour la micro-ROM avec 16 paires d'entrées, l'autre pour la nano-ROM

avec 13 paires d'entrées. Ces deux contrôleurs peuvent être fusionnés en un seul si la topologie le permet. Nous avons aussi rajouté un contrôleur commun aux deux nano-ROMs, ce contrôleur, qui reçoit de l'ordre de 50 paires d'entrées, sera scindé en plusieurs contrôleurs de plus petite taille en fonction du champ de commandes contrôlé ou de la phase de validation des commandes.
La partie opérative a nécessité l'ajoût d'un contrôleur double-rail pour la RAM, un pour l'unité arithmétique, un pour l'unité logique et un pour le contrôle du bus.
Nous verrons par la suite qu'il est aussi nécessaire de rajouter des contrôleurs double-rail dans les interfaces d'entrée et de sortie.
Si les données qui sont envoyées sur le contrôleur n'appartiennent pas à la même phase, il sera nécessaire de compenser le retard par des bascules.

Les contrôleurs double-rails peuvent être implémentés en utilisant des contrôleurs classiques [CAR 68] qui couvrent les pannes simples. Il est cependant plus intéressant, du point de vue topologique et sachant qu'il sera necessaire de les cascader, d'utiliser le contrôleur présenté par le schéma n°10 . Le choix d'utiliser une logique préchargée est justifiée par la plus grande facilité de test d'un telle structure (voir annexes III et IV).

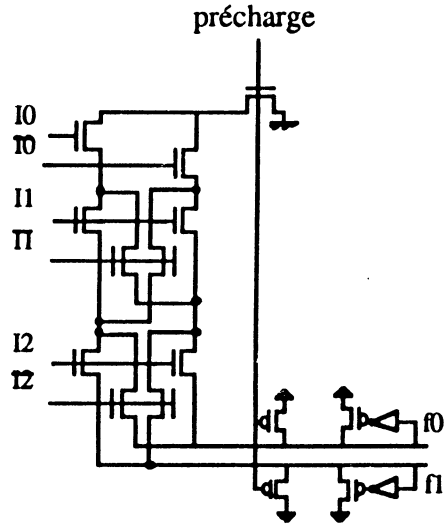
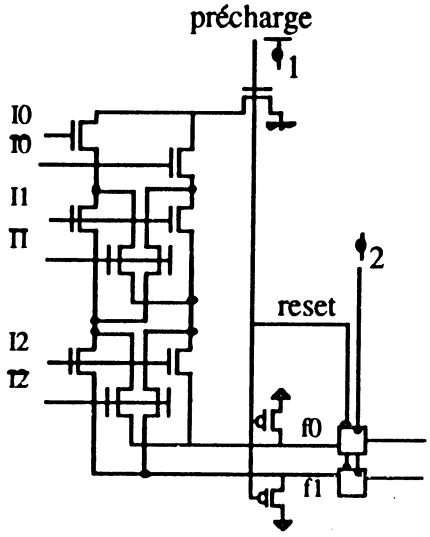


Schéma n°10 : Contrôleur double-rail

Les contrôleurs double-rails des commandes de la partie contrôle (issues du registre de la nano-ROM) possèdent certaines particularités .
Ces contrôleurs se situent après le passage des commandes par la partie opérative, à ce niveau il faut diviser les commandes en trois catégories :
- Les commandes non clockées qui sont contrôlées par le contrôleur décrit par le schéma n°10
- Les commandes clockées sur $\phi 2$ qui sont contrôlées par le contrôleur décrit par la figure de gauche du schéma n°11 .

- Les commandes clokées sur $\phi 3$ qui sont contrôlées par le contrôleur décrit par la figure de droite du schéma n°10 .

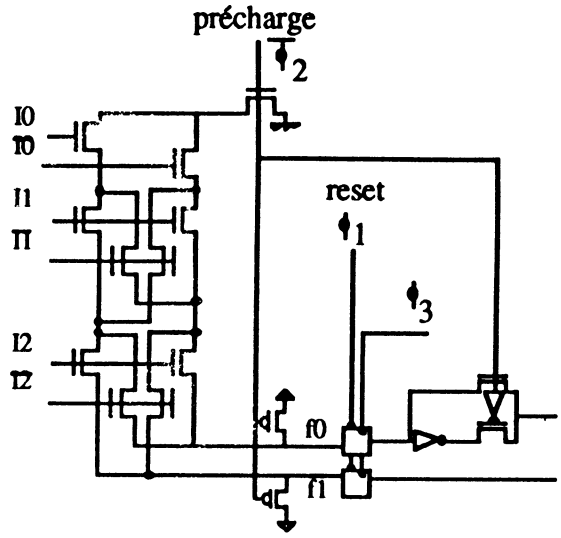
Une commande clockée sur ϕi correspond à un ET logique de la commande et de la phase pour la partie contrôle et à un OU logique de la commande complémentaire et de la phase barre pour la partie contrôle complémentaire.



$\phi 1$barre : précharge et reset des bascules

$\phi 2$: lecture, les signaux sont valides

$\phi 3$: Les entrées changent mais il n'y a pas de lecture. Les mémoires conservent les signaux précédents



$\phi 1$barre : reset des bascules

$\phi 2$barre : précharge et inversion d'une des deux sorties car le mot est hors-code

$\phi 3$: Lecture, les signaux sont valides

Schéma n°11 : Description des contrôleurs double-rails de la partie contrôle

Le reset est utilisé pour éviter que les bascules ne se bloquent définitivement dans un état correct et ainsi rendre inopérationnel le contrôleur double-rail.

Nous interdirons la mémorisation d'erreur en fin de phase $\phi 1$ et, de ce fait il n'est pas nécessaire d'avoir des mots appartenant au code en sortie des contrôleurs double-rails, car cela correspond à la phase de précharge qui met les sorties des contrôleurs double-rails dans une situation d'erreur.

Les sorties des contrôleurs décrits par le schéma n°10 sont valides sur les 3 dernières phases, celles des contrôleurs décrits par le schéma n° 11 gauche sont valides sur $\phi 2$ et enregistrées sur les deux dernières phases. Les sorties des contrôleurs décrits par le schéma n°11 droit ne sont valides que sur $\phi 3$, il est donc préférable que la précharge ait lieu sur $\phi 2$. Par contre cela veut dire qu'au niveau du contrôleur double-rail final il peut y avoir un changement des entrées entre $\phi 2$ et $\phi 3$ ((0,1) --> (1,0) ou inversement) sans qu'il y ait une nouvelle précharge. Pour éviter ce problème il suffit de ne

mémoriser l'erreur qu'à la fin de $\phi 3$ et $\phi 4$ et d'effectuer une précharge du contrôleur double-rail global sur les autres phases. Cela ne pose aucun problème car pour la plupart des contrôleurs, les entrées sont fixes sur les 4 phases et si elles ne le sont pas, on les mémorise dès qu'elles sont valides.

VI.4 LA MEMORISATION D'ERREUR

VI.4 : Rôle

Les signaux d'indication d'erreur globale délivrés par le contrôleur final du schéma n° 1, n'indiquent pas l'occurrence d'une erreur de façon constante. Il est possible qu'après l'apparition de valeurs erronées sur les sorties d'un bloc, il y ait à nouveau des valeurs correctes sur ces mêmes sorties. Dans ce cas, l'erreur sera indiquée pendant un court instant, dont la durée n'est pas suffisante pour déclencher le mécanisme de coupure d'alimentation. La détection d'une panne doit donc être mémorisée de façon permanente. De plus, le mécanisme de mémorisation doit avoir la propriété de détecter ses propres pannes.

VI.4.2 : Application

Le circuit décrit par le schéma n° 12 a la propriété de mémoriser les indications d'erreur appliquées sur ses entrées. Il détecte aussi ses propres pannes mais il ne peut pas assurer la mémorisation de cette détection. On utilise alors 2 de ces circuits, comme l'indique le schéma n° 13 afin que l'un mémorise la détection des pannes de l'autre. Les sorties de ce circuit a0, b0, a1, b1 commandent le passage d'une fréquence qui contrôle l'alimentation externe du circuit. Ces sorties seront détaillées dans le chapitre suivant.

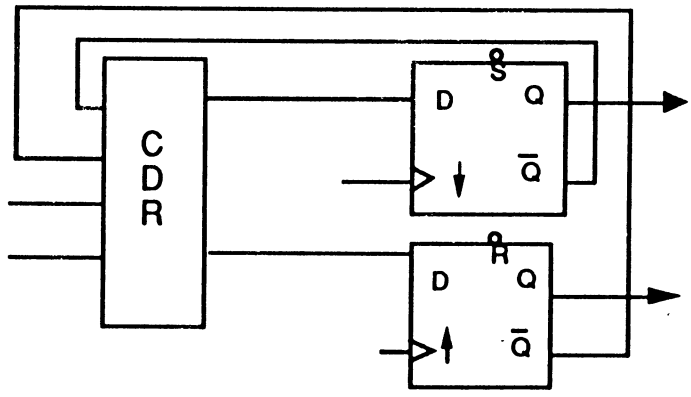


Schéma n° 12 : Mémorisation d'erreur

Le circuit de mémorisation peut être implémenté à partir d'un circuit synchrone (voir schéma n°13). Ce circuit enregistre la présence d'une erreur à la fin de chaque phase (voir schéma n°14) car il se peut qu'une erreur apparaisse sur une phase, puis disparaisse sur la suivante, et ainsi restée indétectable. De plus, comme l'enregistrement des erreurs se fait en fin de phase, toutes les transitions du type 0,1 en 1,0 n'apparaissent pas comme des erreurs.

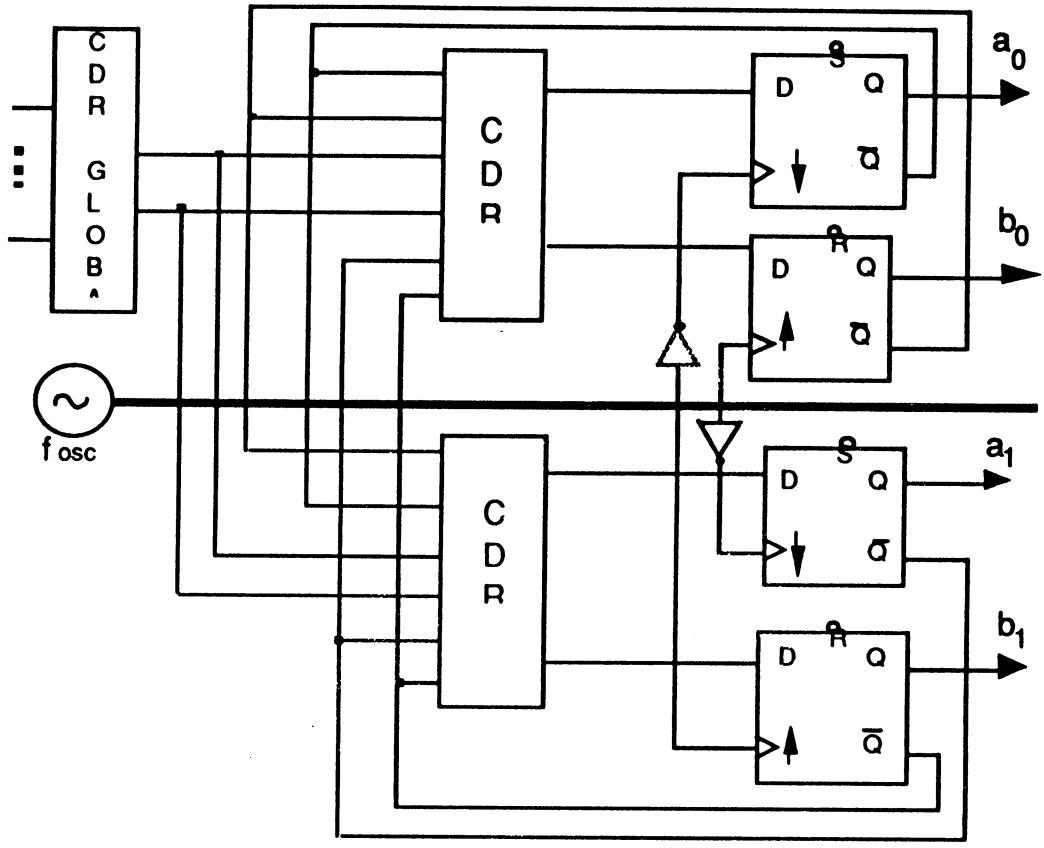


Schéma n° 13 : Circuit d'auto-mémorisation d'erreur

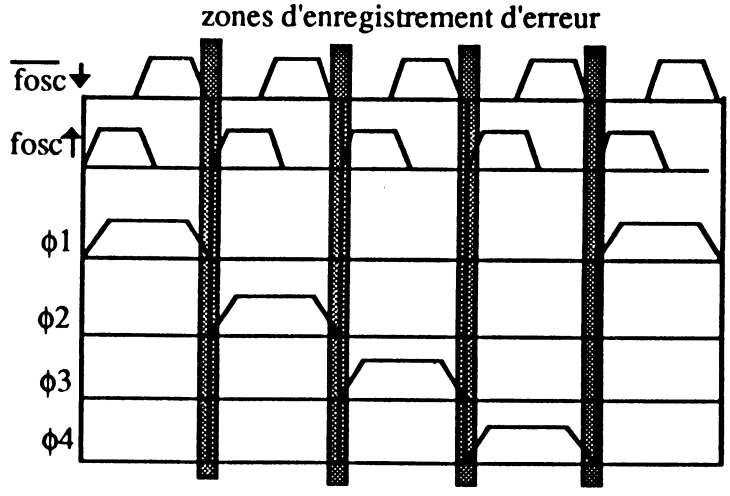


Schéma n°14 : Zones d'enregistrement des erreurs

Nous avons vu, au paragraphe précédent, qu'il était préférable de mémoriser l'erreur à la fin de la phase $\phi 3$ et $\phi 4$. La solution la plus économique pour éliminer les deux phases $\phi 1$ et $\phi 2$ est d'utiliser le petit circuit décrit par le schéma n° 15.

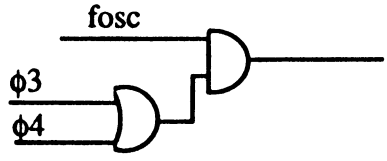


Schéma n° 15 : Signal de mémorisation d'erreur

Du point de vue topologique, il est nécessaire de placer le bloc de mémorisation d'erreur en amont du générateur de fréquences des sorties, ainsi une coupure de la ligne d'horloge interrompra forcément les fréquences de sortie. Le circuit sera alors dans un état sûr. D'autre part les lignes d'horloge des bascules de mémorisation doivent être suffisamment éloignées les unes des autres pour éviter tout court-circuit.

VI.5 CONCLUSION

Ce chapitre vient de décrire les modifications de l'architecture en vue d'obtenir un autotest en-ligne du circuit. Le choix de la modification de l'architecture s'est orienté vers l'utilisation d'un code double-rail et parité (PO) et double-rail et Berger (PC).

Cette solution est très coûteuse en surface puisqu'on la multiplie par plus de 2 (duplication + contrôle) mais elle a l'avantage d'être beaucoup plus facile à valider car les règles de conception ne sont à vérifier qu'entre blocs. Cette solution permet la détection des pannes triples au niveau des blocs mais elle n'assure aucune protection contre les pannes latentes.

Le prochain chapitre fera donc l'objet de l'intégration d'un test hors-ligne pour obtenir une architecture UBIST et ainsi éliminer ce dernier problème.

CHAPITRE VII

IMPLEMENTATION UBIST OU TEST HORS-LIGNE DU CIRCUIT

VII.1 INTRODUCTION

Ce chapitre a pour objectif d'appliquer directement les théories établies au chapitre III au cas particulier du circuit MAPS, c'est à dire de vérifier l'hypothèse H2 . Cette hypothèse, qui ne peut pas être vérifiée dans les conditions normales de fonctionnement, doit cependant être vérifiée si l'on veut garantir la sécurité du circuit. La seule façon d'être sûr que toutes les combinaisons d'entrée ont été appliquées est de les appliquer périodiquement, par logiciel ou par matériel, lors d'une interruption du programme d'application . C'est le test hors-ligne obtenu par la technique BIST.

Comme pour les chapitres précédents nous allons aborder les modifications de chacun des blocs en vue d'obtenir, le plus souvent possible, un test hors-ligne exhaustif.
Nous verrons que ces modifications sont principalement basées sur l'ajout de générateurs de test spécifiques. Cette dernière étape nous permettra ainsi de garantir la très haute sécurité requise pour le circuit.

VII.2 ARCHITECTURE GENERALE

L'intégration d'un test hors-ligne permettant de vérifier l'hypothèse H2 entraîne des modifications d'architecture. Ces modifications sont principalement dues à l'ajout d'un générateur de test capable d'activer l'unité arithmétique, l'unité logique, et le contrôleur double-rail du bus, ainsi que de deux générateurs de test capables d'activer chacune des parties contrôle. D'autre part, le test de la RAM est purement logiciel, il n'influe donc que sur le nombre de lignes du plan mémoire de la partie contrôle. L'activation des générateurs de test et des analyseurs de signature nous oblige à prévoir un champ de test qui augmente la largeur de chacun des plans mémoire. Toutes ces modifications peuvent être résumées par le schéma n°1. Dans la suite de ce chapitre nous nous attacherons à préciser les modifications apportées à chacun des blocs.

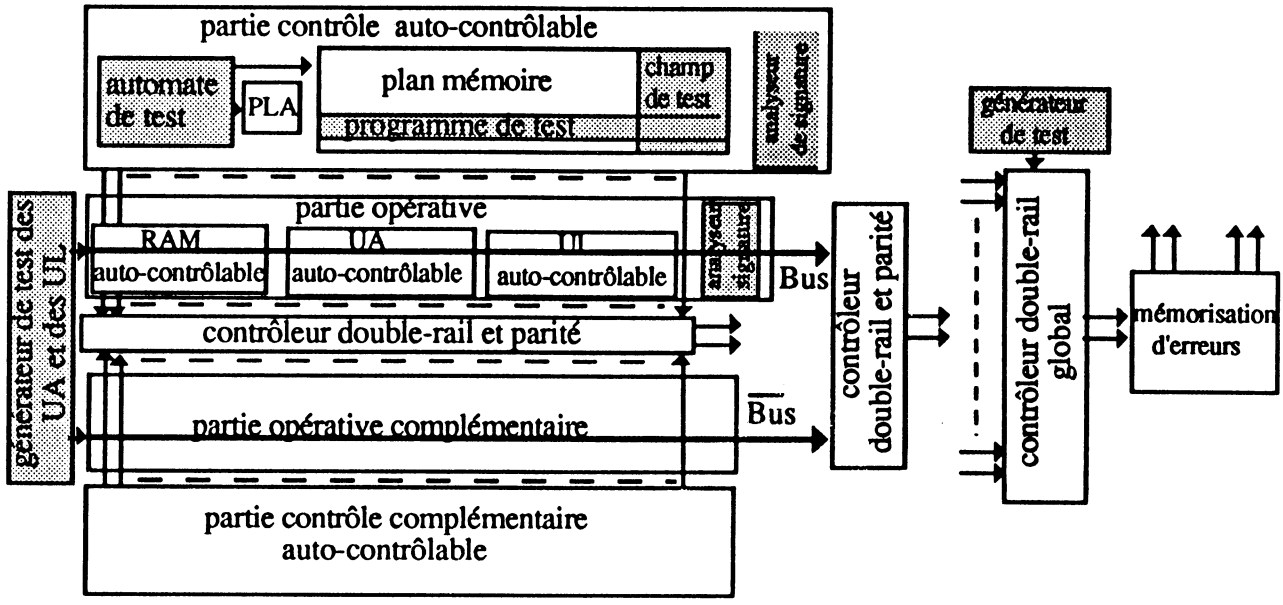


Schéma n° 1 : Description générale de l'architecture du circuit auto-testable

VII.2.1 : La partie opérative

En respectant la même démarche que celle qui a été suivie jusqu'à présent nous allons successivement détailler chacun des blocs de la partie opérative.

VII.2.1.1 : L'unité arithmétique

VII.2.1.1.1 : généralités

Le test des unités arithmétiques peut se faire à l'aide de séquences pseudo-aléatoires [THO 87], mais cette technique nécessite de très longues séquences et n'assure pas une couverture de panne complète. D'autres auteurs [CER 88] ont proposé d'utiliser la technique BIST pour une architecture particulière [ABO 84]. La technique que nous proposons repose sur l'implantation de générateurs de test basés sur la testabilité C [FRI 73], [SRI 81]. Dans ce cas, la complexité des générateurs et la taille des séquences de test sont indépendantes du nombre d'entrées de l'unité arithmétique. Ce qui veut dire qu'une unité arithmétique peut être testée si l'on trouve la plus petite séquence qui permet d'exercer exhaustivement chacune des tranches. Il suffit ensuite de l'appliquer simultanément sur chacune des tranches.

De manière générale, si l'on applique la même séquence à deux tranches consécutives, cette seconde tranche ne reçoit pas toutes les combinaisons d'entrée.

Par contre on peut facilement vérifier par le calcul que si :

f { (Co,a1,b1) ---> (i,j,k), $\forall$(i,j,k)$\in$ E^3 = (0,1)x(0,1)x(0,1)}est une bijection

alors g{ (C1,a1$\oplus$Co,b1$\oplus$Co) ---> (i,j,k) $\forall$(i,j,k)$\in E^3$ = (0,1)x(0,1)x(0,1)} avec C1 = a1.b1 + Co(a1 $\oplus$b1) est aussi une bijection.

En supposant que si

fn { (Co,a1,b1.....ai,bi.....an,bn) ---> (i,j1,k1.....ji,ki.....jn,kn), $\forall$(i,.....jn,kn)$\in$ E^{2n+1} } est une bijection

alors gn{ (Cn,a1$\oplus$Co,b1$\oplus$Co.....an$\oplus$Co,bn$\oplus$Co) ---> (i,.....jn,kn) $\forall$(i,.....jn,kn)$\in E^{2n+1}$ } est aussi une bijection.

Démontrons que

fn+1{(Co, a1, a2.....ai, bi.....an, bn, an+1, bn+1) ---> (i, j1, k1.....ji, ki.....jn, kn, jn+1, kn+1), $\forall$(i,.....jn,kn,jn+1,kn+1)$\in$ $E^{2(n+1)+1}$ } est une bijection

alors gn+1{ (Cn+1,a1$\oplus$Co,b1$\oplus$Co.....an$\oplus$Co,bn$\oplus$Co,an+1$\oplus$Co,bn+1$\oplus$Co) ---> (i,.....jn,kn,jn+1,kn+1) $\forall$(i,.....jn,kn,jn+1,kn+1)$\in E^{2(n+1)+1}$ } est aussi une bijection.

fn+1 est une bijection puisque l'on applique systématiquement à chaque entrée toutes les combinaisons.

Deux cas peuvent alors se produire

Si Cn = Cn+1 => 1 = an+1.bn+1 + an+1$\oplus$bn+1

ou 0 = an+1.bn+1

qui donne les solutions (Cn,Cn+1, an+1$\oplus$Co, bn+1$\oplus$Co) = (0,1,1,1) ou (1,0,0,0)

Si Cn $\neq$ Cn+1 => 1 = an+1.bn+1

ou 0 = an+1.bn+1 + an+1$\approx$bn+1

qui donne les solutions (Cn,Cn+1, an+1$\oplus$Co, bn+1$\oplus$Co) = (0,0,0,1) , (0,0,0,0), (0,0,1,0) , (1,1,1,1),(1,1,0,1) , (1,1,1,0),

On obtient tous les [2(n+1) +1]-uplets suivants (0,.....,0,1), (0,.....,1,0), (1,.....,0,1),(1,.....,1,0),(0,.....,0,0),(0,.....,1,1),(1,.....,0,0) , (1,.....,1,1)

Sachant que, par hypothèse, tous les éléments intermédiaires appartiennent à une bijection et qu'ils sont associés aux 8 triplets uniques, on peut conclure que gn+1 est aussi une bijection.

Proposition

Si les entrées (Co, a1,.....,an,bn) de chaque ensemble de k tranches consécutives d'une unité arithmétique sont exhaustivement exercées alors on assure la détection de

toutes les combinaisons de panne, tel qu'il y a moins de r cellules consécutives en panne avec $1 \leq r \leq k$ pour lesquelles la première cellule suivant et la première cellule précédent cette serie soient sans panne .

La démonstration de cette proposition a été présentée dans [NIC 90a].

en conséquence on peut affirmer que le générateur du schéma n° 2 permet de détecter toutes les pannes simples et doubles puisque l'on teste exhaustivement tous les couples de deux tranches successives.

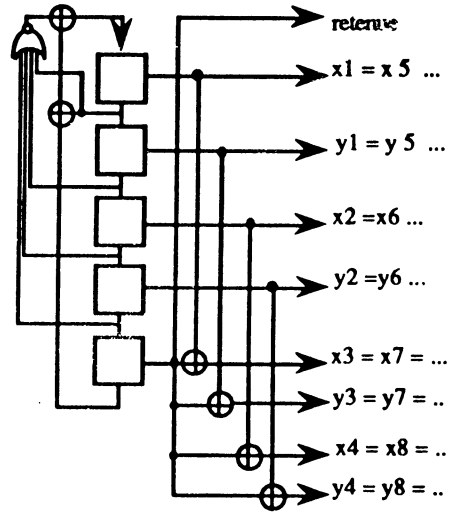


Schéma n°2 : Générateur de test pour une unité arithmétique

La couverture des pannes triples est donnée par la formule suivante :

$$\frac{\text{nombre de pannes non couvertes}}{\text{nombre de pannes possibles}} = 1 - \frac{(n-2)}{C_n^3 + 2C_n^2 + C_n^1}$$

Soit 95% pour n = 8 . Le numérateur (n - 2) représente le nombre de pannes triples survenant sur trois cellules consécutives et donc non couvertes. La formule peut être étendue à la couverture des pannes de multiplicité supérieure.

Les unités arithmétiques que nous utilisons sont conçues en CMOS complémentaire, elles sont donc sensibles aux pannes du type S-open qui leur confèrent un comportement séquentiel. La détection de ces pannes nécessite l'utilisation de deux séquences de test, l'une pour initialiser et l'autre pour tester [WAD 78]. En règle générale, il faut donc utiliser un LFSR disposant de 2 fois plus de cellules.

VII.2.1.1.2 : Application

Pour des raisons que nous exposerons ultérieurement, nous disposons d'un LFSR de 16 cellules pour le test du contrôleur double-rail du bus. Ce générateur peut aisément être reconfiguré en deux générateurs de 8 cellules, l'un pour l'unité arithmétique de la partie opérative et l'autre pour l'unité arithmétique de la partie opérative duale. Grâce à ce LFSR de 8 cellules nous pouvons tester exhaustivement chaque paire de cellules (taille minimum du LFSR 5 cellules voir schéma n°2) et l'on peut tester les S-open pour chaque tranche (taille minimum du LFSR 6 cellules). Le test des S-open de chaque paire de tranche aurait demandé un LFSR de 10 cellules. Les pannes simples et doubles sont donc couvertes à 100% (95% pour les pannes triples).

Le générateur du micro-contrôleur MAPS est décrit par le schéma n°3.

A partir de ce schéma, nous pouvons remarquer que la première tranche reçoit :

(a1, b1, Co) = (A2, A4, A7)

La seconde recoit :

(a2, b2, C1) = (A6, A8, C1) avec C1 = a1.b1 + a1.Co + b1.Co

Chaque paire de tranches est donc testée exhaustivement puisque toutes les combinaisons d'entrées et toutes les propagations de retenue sont présentes. La troisième et la quatrième tranches reçoivent le X-OR de la retenue entrante Co et des vecteurs d'entrée des deux premières tranches.

On peut aussi remarquer que la première tranche reçoit le triplet (a1, b1, Co) puis le triplet ($a1^-$, $b1^-$, a2).

La seconde tranche reçoit le triplet (a2, b2, C1) puis le triplet ($a2^-$, Co, $C1^+$) avec $C1^+ = a1^-.b1^- + a1^-.a2 + b1^-.a2$.

A partir de ces données, on peut vérifier que chaque tranche reçoit donc tous les couples d'entrée possibles.

On a pu vérifier que les séquences ainsi produites et appliquées à l'unité arithmétique du schéma n°4 du chapitre précédent sont robustes car on peut fixer deux entrées sensibilisant un chemin et ne faire varier que les deux autres. Seule, l'une de ces deux entrées permet de faire varier la sortie choisie. Il n'y a donc pas de problème d'aléas.

Au temps t du cycle i, le générateur de test envoie 8 entrées dans l'accumulateur A par l'intermédiaire du bus sur $\phi 3$. Au cycle i+1 le générateur envoie 8 autres entrées (prises sur des cellules différentes) dans le registre B sur $\phi 2$ et l'opération d'addition s'effectue sur $\phi 3$. Le générateur de test doit donc fonctionner à une fréquence deux fois moins importante soit 500KHz.Il est donc nécessaire de prévoir un multiplexage de l'horloge du générateur de test, la commande de ce multiplexeur sera assurée par les commandes de test TESTUAX et TESTUAY.

Sur les 8 entrées que reçoit chaque registre, les 4 premiers bits sont égaux aux 4 derniers, dans ce cas,la parité est toujours nulle.

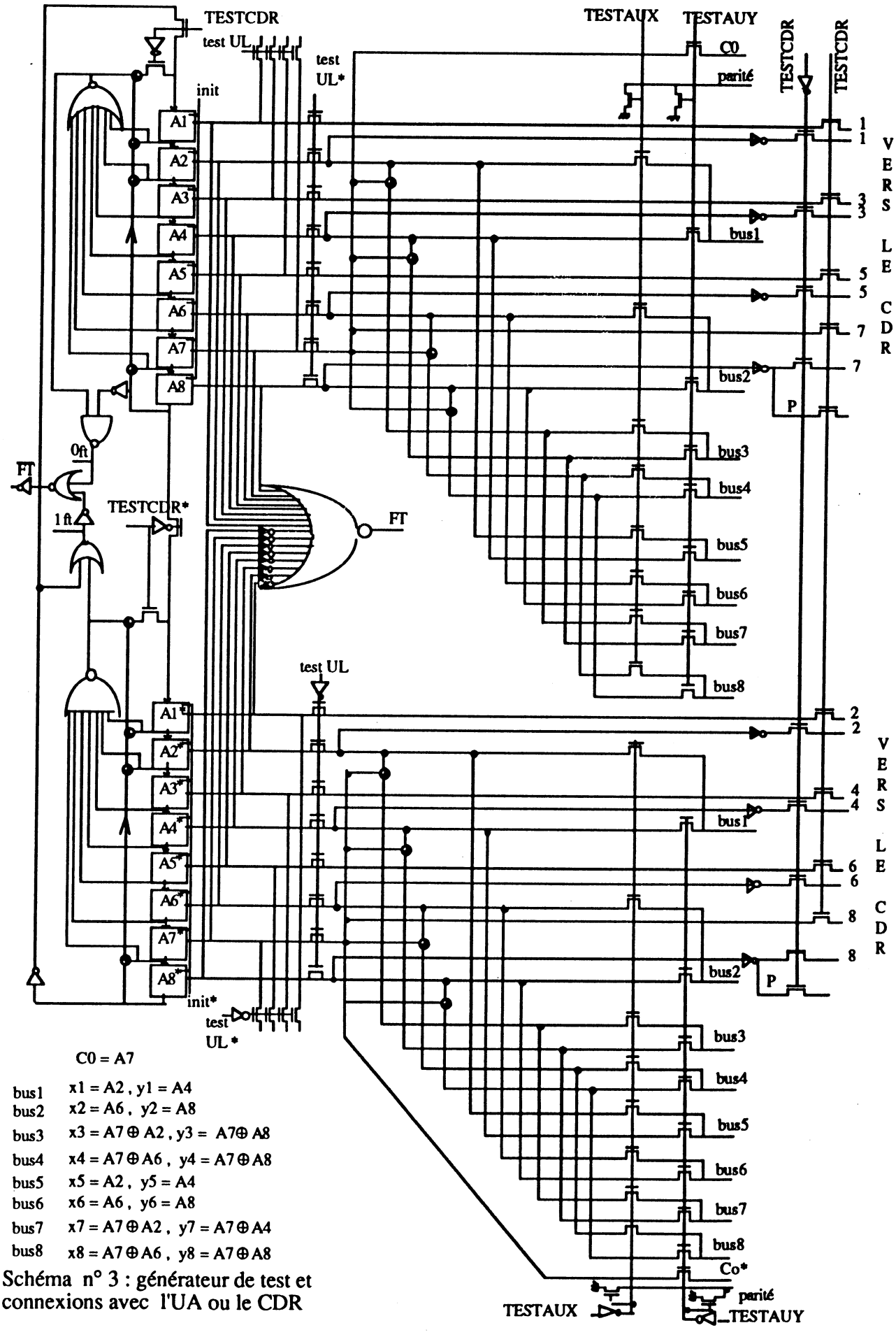


Schéma n° 3 : générateur de test et connexions avec l'UA ou le CDR

L'additionneur est testé mais les commandes d'inversion et de passage du zéro ne le sont pas.Comme il ne peut pas être question d'utiliser un autre générateur, le test se fera par logiciel. Le programme de test ainsi créé sera appliqué successivement à A puis à B. Ce programme teste les S-opens du NAND et de l'inverseur du schéma n°8 du chapitre V . Le programme dual doit quant à lui tester les S-opens des NOR et des inverseurs présentés par le schéma n° 4.

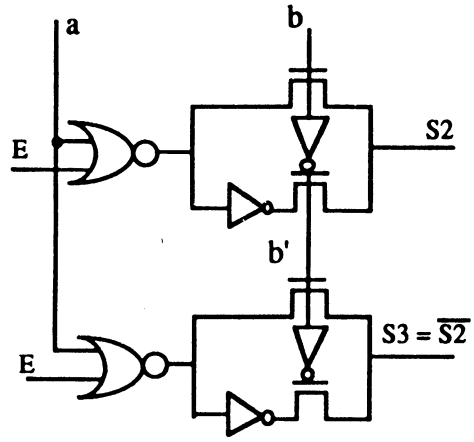


Schéma n°4 : Fonctionnement de la mise à 0 et de l'inversion

partie opérative					partie opérative complémentaire						
E	a	b	S'	S1	E	a	b	S'	S2	b'	S3
1	1	1	0	0	0	0	0	1	0	1	1
0	1	1	1	1	1	0	0	0	1	1	0
1	1	1	0	0	0	0	0	1	0	1	1
1	0	1	1	1	0	1	0	0	1	1	0
1	0	0	1	0	0	1	1	0	0	0	1
1	1	0	0	1	0	0	1	1	1	0	0
0	1	0	1	0	1	0	1	0	0	0	1

Tableau n° 1 : Séquence de test de la mise à 0 et de l'inversion

VII.2.1.2 : L'unité logique

Le test de l'unité logique peut être effectué avec le même générateur que celui que l'on vient de présenter, mais en utilisant une nouvelle configuration. Pour cela, on connecte les cellules paires du générateur, à l'entrée de C, grâce au multiplexage 4 --> 1, à l'entrée de C', en ajoutant un nouveau multiplexeur 2 --> 1, à la place de la commande de sélection du OU ou du ET, en ajoutant

un multiplexeur 2 --> 1, et enfin à la place de la commande de la boîte d'inversion en ajoutant un multiplexeur 2 -->1.

La technique utilisée est la même que celle employée précédemment sauf que dans ce cas il n'y a qu'une seule tranche à tester. Le test sera effectué une fois en utilisant le registre d'entrée C et une seconde fois en utilisant le registre d'entrée C".

On assure ainsi la détection de toutes les pannes simples et de tous les S-opens.

Comme cela était déjà le cas avec l'unité arithmétique, il nous reste à tester les lignes de commande qui permettent, soit de laisser passer la valeur de C ou C', soit d'imposer un zéro à leurs sorties. Cela peut être fait par microprogramme.

VII.2.1.3 : La RAM

VII.2.1.3.1 : Généralités

La RAM peut détecter en ligne les erreurs, mais comme certaines cases mémoire ne sont que très peu utilisées, il se peut qu'au cours du temps certaines pannes latentes surviennent et que lorsque exceptionnellement on utilise cette case, la valeur qu'elle fournit est erronée mais appartient au code. Dans ce cas l'erreur est indétectable, ce cas est cependant très rare puisqu'il faut que les deux RAMs aient les mêmes pannes latentes pendant la même période.

Pour diminuer encore les risques d'erreur on a implanté dans la micro-ROM un algorithme de test de la RAM. Cet algorithme est l'algorithme de MARINESCU modifié [MAR 82] (voir tableau n°2) qui permet de détecter tous les collages et toutes les influences idempotentes de deux cellules quelconques.

n° cel	init	séquence S1	séquence S2	séquence S3	séquence S4
1	0	R0↑↓↑	R1↑R0↓R1	R1↑↓↑	R0↑R1↓R0
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
k	0	R0↑↓↑	R1↑R0↓R1	R1↑↓↑	R0↑R1↓R0
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
n	0	R0↑↓↑	R1↑R0↓R1	R1↑↓↑	R0↑R1↓R0

On applique la séquence S1 sur toutes les cellules, puis S2, puis S3 et S4. La séquence S1 et S2 incrémentent les adresses, S3 et S4 les décrémentent.

Tableau n°2 : Algorithme de MARINESCU

Une influence est idempotente si un changement d'une cellule i de 1 à 0 respectivement de 0 à 1 modifie le contenu d'une autre cellule j de 1 à 0 ou de 0 à 1 et si une nouvelle transition identique de la cellule i survient, alors la cellule j n'est pas modifiée.
Le contraire d'une influence idempotente est une influence d'inversion, ces influences ne sont pas couvertes.

VII.2.1.3.2 : Application

Pour adapter l'algorithme de MARINESCU il a fallu apporter quelques modifications car contrairement à l'algorithme original nous ne voulons en aucun cas perdre le contexte du circuit. Pour cela, la phase initiale de mise à 0 est supprimée (de toute façon, elle ne détecte aucune panne). D'autre part, on remarque que chaque séquence de lecture-écriture d'un mot de la RAM débute par une lecture. On stocke le résultat de cette lecture dans le registre B et si l'algorithme de MARINESCU indique l'écriture d'un 0, on écrit le contenu du registre B, sinon on écrit son inverse. Cette modification aura pour conséquence d'échanger des écritures à 1 par des écriture à 0 et inversement, mais en observant la parfaite symétrie de l'algorithme, de sa démonstration et du modèle de panne considéré, on peut constater que la couverture de panne n'est pas affectée. Lorsque l'algorithme est terminé, ce sont les valeurs initiales de la mémoire qui sont restaurées.
C'est à partir de ce registre B que l'on effectue ensuite toutes les opération de lecture, d'écriture et d'inversion. Le registre A est utilisé pour incrémenter l'adresse. Cet algorithme ne nécessite aucun registre supplémentaire. Cependant l'algorithme demande d'inverser les 9 bits du mot, le 9ième bit étant la parité, son inversion en même temps que celle de tous les autres bits crée un mot hors-code dont il faut signaler la présence au contrôleur double-rail et parité du bus. Le mécanisme d'inversion de la parité est présenté par le schéma n°5 , nous verrons ultérieurement, lors de la présentation du contrôleur double-rail comment il est prévenu de la présence d'un mot hors-code.

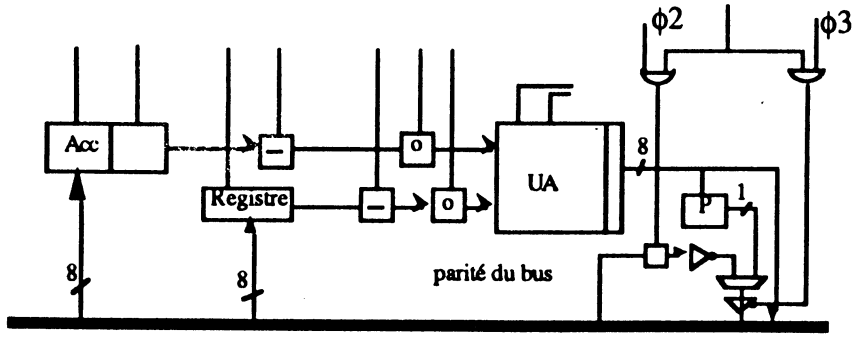


Schéma n°5 : Principe d'inversion de la parité dans l'algorithme de Marinescu

Notons que l'algorithme de MARINESCU est donné pour le test des RAM de 1 bit. Pour qu'il soit valide pour le test des RAMs contenant des mots de plusieurs bits, il faut supposer qu'il n'existe pas d'influence entre les cellules d'un même mot. Pour cela nous avons délibérément choisi une RAM qui possède un décodeur colonne (2 bits) et un décodeur ligne (4 bits), en conséquence les cellules d'un même mot deviennent donc topologiquement éloignées les unes des autres.

VII.2.1.4 : Le contrôleur double-rail du bus

Les contrôleurs double-rails doivent indiquer la présence d'une erreur à la fois lors du fonctionnement normal (test en ligne) et lors du test hors-ligne.
Tous les contrôleurs que l'on utilise assure une détection totale de toutes les pannes simples [CAR 68] et une très grande détection des pannes multiples [NAN 88]. Ceci représente donc une très bonne couverture de panne. Cependant, certains contrôles tels que le contrôle des données transitant par le bus, le contrôle des sorties, et le contrôle global jouent un rôle primordial vis à vis de la sécurité. Pour ces contrôleurs, il est intéressant de couvrir toutes les pannes multiples. Ceci est possible si on applique des vecteurs hors-code lors du test [NIC 88], [NAN 89]. Des générateurs de test spécifiques sont donc associés à chacun de ces contrôleurs.

Le contrôleur double-rail final est un contrôleur self-exercising dont le schéma a été présenté au chapitre III . La couverture de panne répond donc au théorème n°1 du chapitre III .
Les contrôleurs double-rails du bus et des sorties sont identiques, ils sont testés par le générateur présenté par le schéma n°3. Le test de ces contrôleurs est basé sur une approche analytique, c'est à dire que les différentes erreurs sont examinées de manière analytique pour pouvoir élaborer une séquence déterministe de vecteurs de test. Pour obtenir les séquences de test désirées, le générateur du schéma n°3 est reconfiguré en un générateur de 16 cellules , la seizième cellule est réintroduite dans la première après passage par un inverseur. Ce générateur est initialisé à 0 et fournit les séquences du tableau n°3.

Le contrôleur double-rail est conçu avec trois étages comme le montre le schéma n°7. En fonctionnement normal, ce contrôleur ne reçoit que des mots du code avec leurs parités . En mode test chacun des blocs doit recevoir tous les vecteurs hors-code. La seule façon de parvenir à ce résultat est de donner des valeurs aux deux parités qui ne sont plus les parités des mots générés mais des valeurs qui permettent d'obtenir, en entrée du dernier étage du contrôleur, tous les mots hors-code nécessaires au test.

Cependant, ces deux contrôleurs double-rail sont connectés différemment puisque le contrôleur double-rail du bus est relié directement au générateur de test tandis que celui des sorties est relié au générateur de test par l'intermédiaire des registres de sortie. Cette situation est décrite par le schéma n°6 .

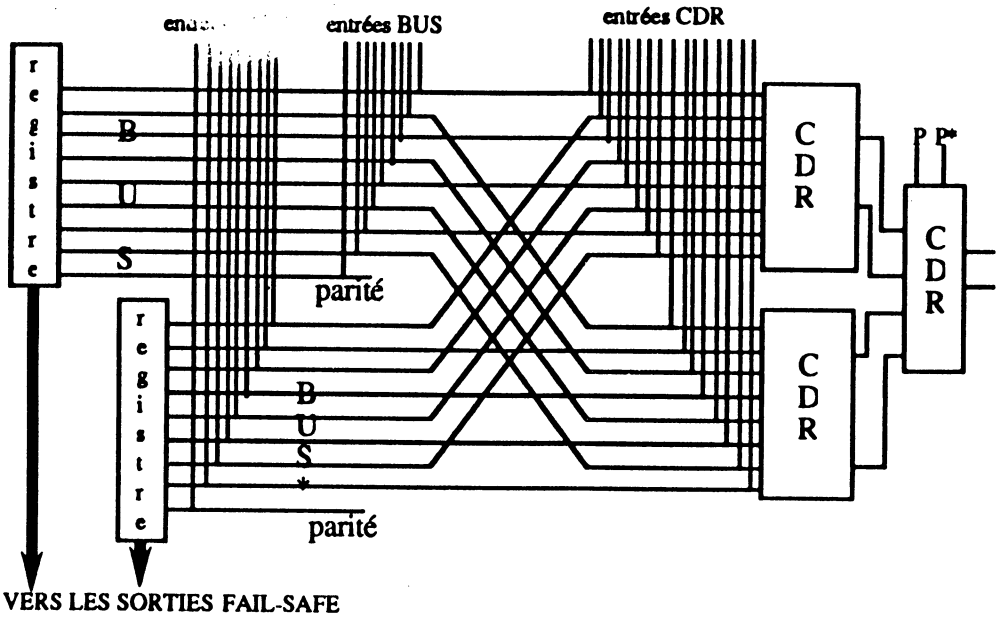


Schéma n°6 : Connexions du générateur de test avec le Bus

Le premier étage reçoit donc tous les vecteurs qui permettent d'assurer la propriété "strongly code disjoint" pour les pannes simples et multiples [NIC 84 b]. Pour que le contrôleur soit "strongly code disjoint" il faut aussi que les étages suivants soient "strongly code disjoint", cela peut être réalisé en utilisant un petit automate qui gère les entrées du dernier étage du contrôleur. Cet automate est décrit par le schéma n°8.

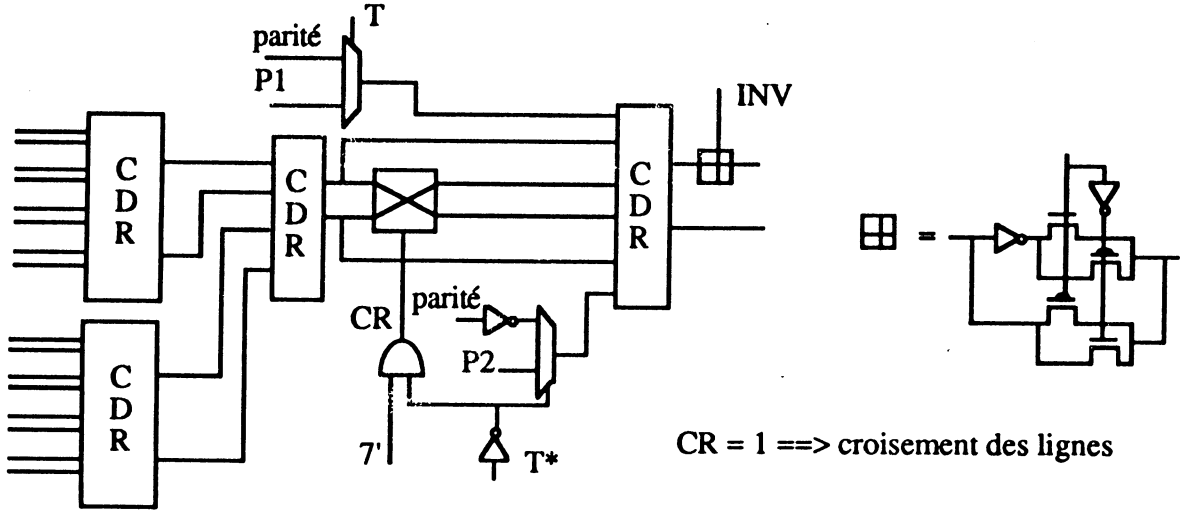


Schéma n°7 : Contrôleur double-rail et ses signaux d'inversion

Cet automate reçoit, comme entrées, les signaux T et T* et la valeur de la cellule 8' du tableau n°3 et produit, en sortie, les signaux INV (INV =1 ==> inversion de la sortie finale du contrôleur) et les deux valeurs P1 et P2 qui remplacent les parités en mode test.Ces deux valeurs permettent d'imposer

les valeurs de test désirées en entrée du dernier étage du contrôleur. Lorsque (T, T*) = (1,0) les valeurs P1 et P2 sont imposées, c'est le mode test.

1	1'	3	3'	5	5'	7	7'	2	2'	4	4'	6	6'	8	8'	entrées étage 2	entrées étage 3	INV	CR
0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1010	010101	0	1
0	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1000	100001	1	1
0	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1001	101011	1	1
0	1	0	1	0	1	0	1	0	1	0	0	1	0	1	0	1000	100001	1	1
0	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0	1010	010100	1	1
0	1	0	1	0	1	0	1	0	1	0	1	0	0	1	0	1000	100001	1	1
0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1001	101010	0	1
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0	1000	100001	1	1
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1010	010101	0	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1110	011110	1	1
1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0110	001010	1	1
1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	1110	011110	1	1
1	0	1	0	0	1	0	1	0	1	0	1	0	1	0	1	1010	110101	1	1
1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	1110	011110	1	1
1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	1	0110	101010	0	1
1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	1110	011110	1	1
1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1010	011001	0	0
1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	1011	011110	1	0
1	0	1	0	1	0	1	0	1	0	0	1	0	1	0	1	1001	000110	1	0
1	0	1	0	1	0	1	0	1	0	1	1	0	1	0	1	1011	011110	1	0
1	0	1	0	1	0	1	0	1	0	1	0	0	1	0	1	1010	111001	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1	1011	011110	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1	0	0	1	1001	100110	0	0
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	1011	011110	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1010	011001	0	0
0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0010	100001	1	0
0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0110	100111	1	0
0	1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	0010	100001	1	0
0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1010	011000	1	0
0	1	0	1	0	0	1	0	1	0	1	0	1	0	1	0	0010	100001	1	0
0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	0110	100110	0	0
0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	0100	100001	1	0

Tableau n°3 : Séquence de test du contrôleur double-rail

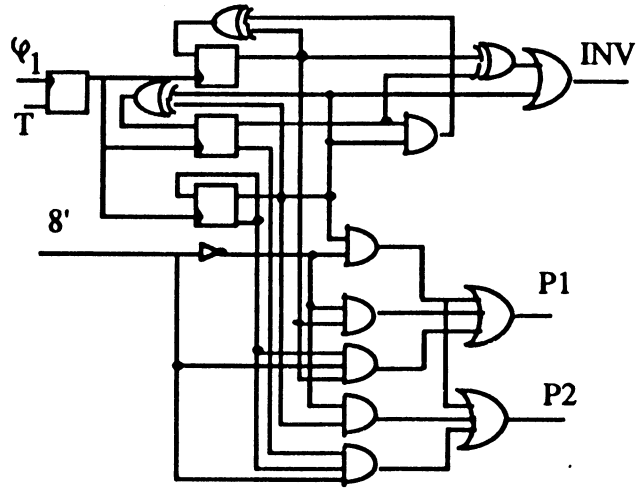
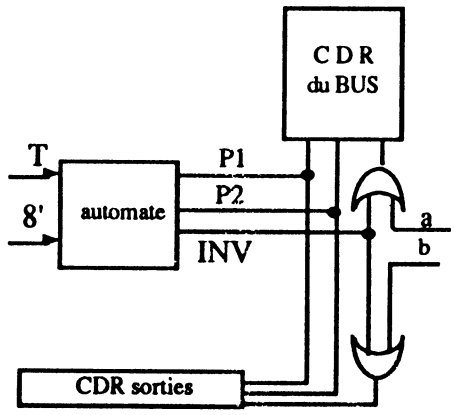


Schéma n° 8 : Automate de séquencement des inversions

Les vecteurs de test, via les registres de sorties et les commandes de croisement et d'inversion, ainsi que les valeur P1 et P2 sont envoyées au contrôleur double-rail des sorties . Ceci permet également de disposer d'un contrôleur double-rail "strongly code disjoint" au niveau des sorties.

Nous avons vu précédemment que lors du test de la RAM certain mots étaient hors-code et qu'il fallait prévenir le contrôleur. Pour résoudre ce problème, il est nécessaire d'associer au circuit du schéma n°5 les modifications présentées au schéma n°9 . a et b sont des commandes générées par la partie contrôle



lors du test des CDR du bus et des sorties
automate actif et (a, b) = (0, 0)

lors de la génération des séquences de test des sorties
automate non actif et (a, b) = (0, 0) si le mot ∈ au code
et (a, b) = (1, 1) si le mot ∉ au code

lors du test de la RAM
automate non actif et (a, b) = (0, 0) si le mot ∈ au code
et (a, b) = (1, 0) si le mot ∉ au code

fonctionnement normal ou lors du test des autres blocs
automate non actif et (a, b) = (0, 0)

Schéma n°9 : Description du fonctionnement du circuit indicateur d'erreur suivant le test en cours

VII.2.2 : La partie contrôle (ROM - PLA)

VII.2.2.1 : Fonctionnement du test

Le test de la partie contrôle est assuré par un automate qui gère successivement le test de la micro - ROM et du PLA. Cet automate (encadré en pointillés sur le schéma n° 10) reçoit

l'autorisation de début de test de la partie contrôle (DT , DT*), il fonctionne alors indépendamment de la partie contrôle, en fin de test il redonne la main à la partie contrôle.

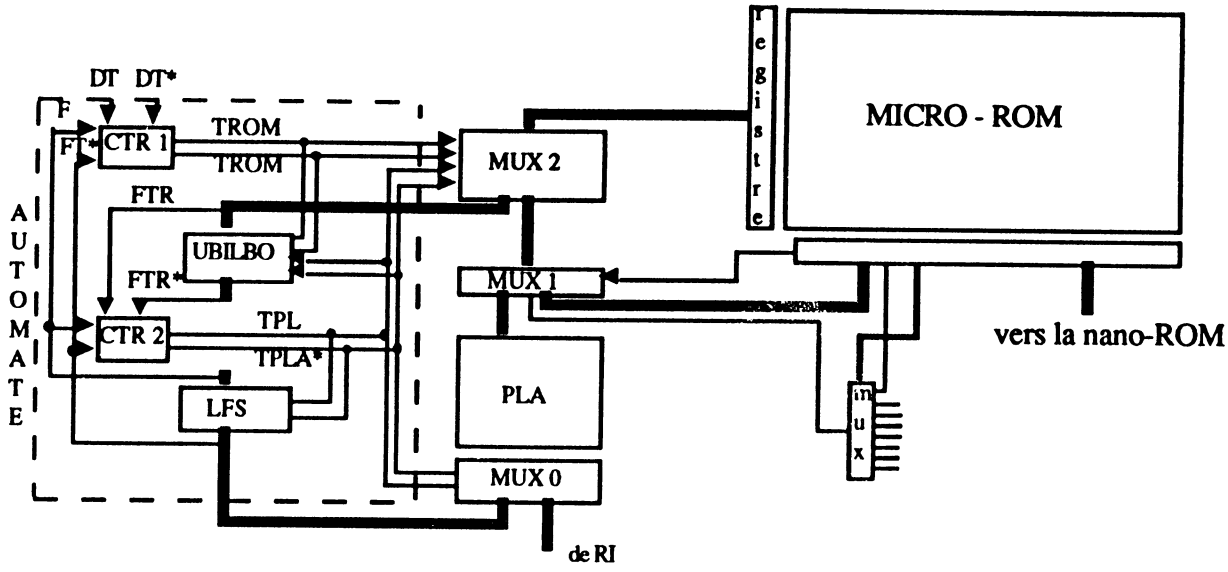


Schéma n° 10 : Détail du test hors-ligne de la partie contrôle

En fonctionnement normal, les adresses de la micro - ROM proviennent soit du PLA, soit de la micro - adresse suivante. La sélection est assurée par un bit de la micro - commande.
En phase de test (qui débute par le test de la partie contrôle) DT et DT* sont activés une fois pour donner la main à l'automate. Dans un premier temps, l'automate effectue le test de la micro - ROM, pour cela, il déclenche le UBILBO et contrôle le multiplexeur 2 avec les commandes TROM , TROM* et TPLA , TPLA*.

Les commandes TROM et TROM* sont aussi utilisées pour désactiver certaines commandes susceptibles de détruire le contenu des organes de mémorisation (commande d'écriture de la RAM ou des registres de sortie). Lorsque la séquence de test est totalement déroulée, un compte-rendu FTR est envoyé au deuxième contrôleur de l'automate (CTR 2) . Ce contrôleur déclenche le LFSR et sélectionne MUX 0 pour que les adresses générées par le LFSR soient connectées au PLA. Le MUX 1 doit sélectionner les adresses venant du PLA. Cette sélection peut être assurée par les commandes TPLA et TPLA* (voir schéma n° 11).

Il sera intéressant de bien choisir la dernière micro-adresse générée par le UBILBO lors du test de la micro-ROM car cela peut nous permettre de déclencher le test d'un autre organe du circuit sans que l'on ait besoin d'un compte-rendu de fin de test de la PC. Ce test doit bien évidemment être plus court que celui du PLA. Cette adresse doit aussi imposer au MUX1 le passage des adresses venant du PLA car on veut tester celui-ci.

Les adresses venant du LFSR sont envoyées sur le PLA, elles traversent le MUX 1, puis elles entrent dans le MUX 2 et sont dirigées vers le UBILBO reconfiguré en analyseur de signature. Ce chemin particulier et la reconfiguration sont assurés par les commandes TPLA et TPLA*.
Dès que le LFSR a fini de dérouler sa séquence de test, un signal de compte-rendu (FT) est envoyé aux deux contrôleurs CTR 1 et CTR 2. CTR 1 redonne la main à la PC pour poursuivre le test, CTR 2 bloque le LFSR et change la sélection de MUX 0 et MUX 2. FT est repositionné à 0 pour préparer le prochain test au prochain STOP.

L'automate présenté est totalement dupliqué, chacun d'eux recevant des données complémentées (DT, DT*) de sa partie contrôle respective.

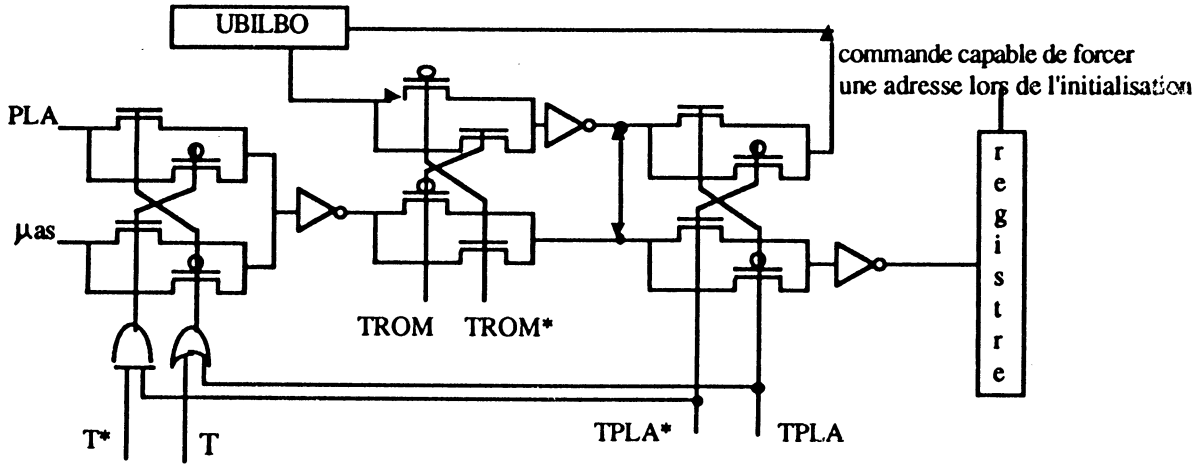


Schéma n° 11 : Multiplexage à 2 niveaux

VII.2.2.2 : Détection d'erreurs

Comme les PLAs sont testés grâce à l'analyse de signature, il faut donc que le résultat soit conservé dans les bascules du UBILBO. La réinitialisation ne doit intervenir que lorsque la signature aura été lue. Les détails concernant le UBILBO et le LFSR sont donnés dans les schémas n°12 et n°13.

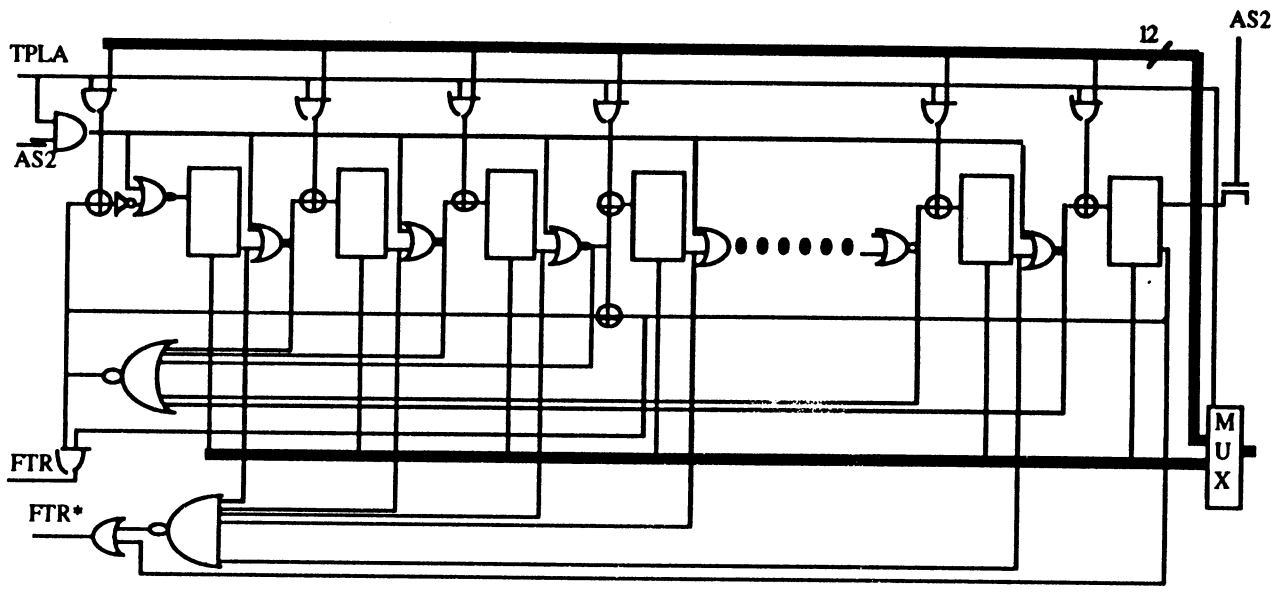


Schéma n° 11 : Partie BILBO du UBILBO

La phase de test débute par le test de la micro-ROM c'est à dire (TROM, TPLA, AS2) =(1, 0, 0), le BILBO est alors configuré en un LFSR de 12 cellules. La seconde phase permet de tester le PLA, dans ce cas (TROM, TPLA, AS2) =(0, 1, 0), le BILBO devient analyseur de signature (entrée des données par le haut du schéma). Lorsque l'on veut lire la signature on impose AS2 = 1 ((TROM, TPLA,) =(0, 0) puisque le test en terminé), le BILBO devient registre à décalage. En fin de lecture de signature le BILBO ne contient que des 0, au cas ou l'initialisation devrait être différente de 0 on utiliserait de signal DT pour initialiser, juste avant d'effectuer le test de la micro-ROM.

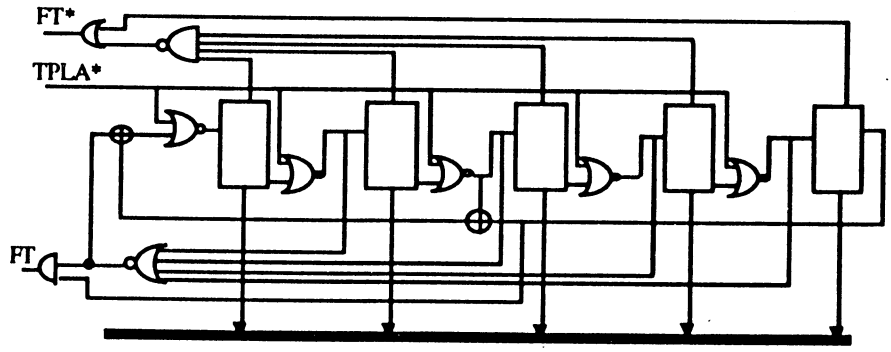


Schéma n° 13 : LFSR

Les automates de contrôle CTR1 et CTR2 sont identiques (voir schéma n° 14), seuls les signaux qui les déclenchent sont différents. Une petite modification de l'automate CTR2 permettrait de générer deux fois la séquence du LFSR de telle sorte que l'on analyse sa signature, la première fois et ainsi

conserver l'avantage de pouvoir effectuer un autre test en parallèle et que l'on verifie son code, par traversée de la ROM, la seconde fois.

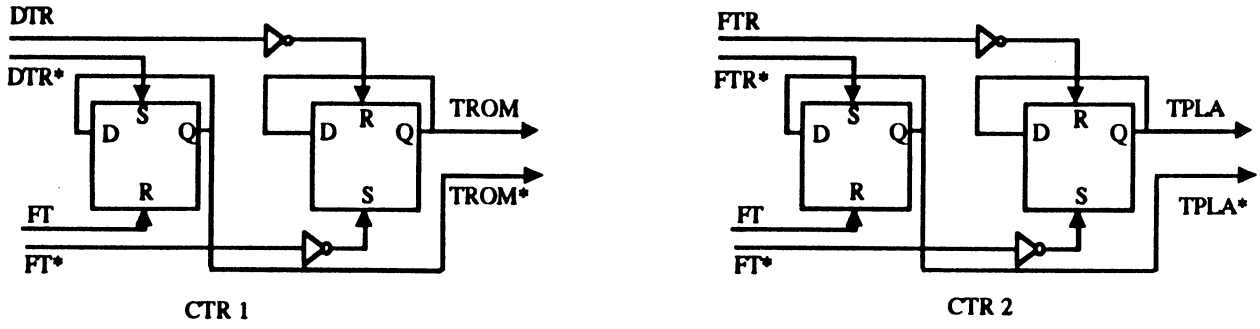


Schéma n° 14 : Automates de contrôle

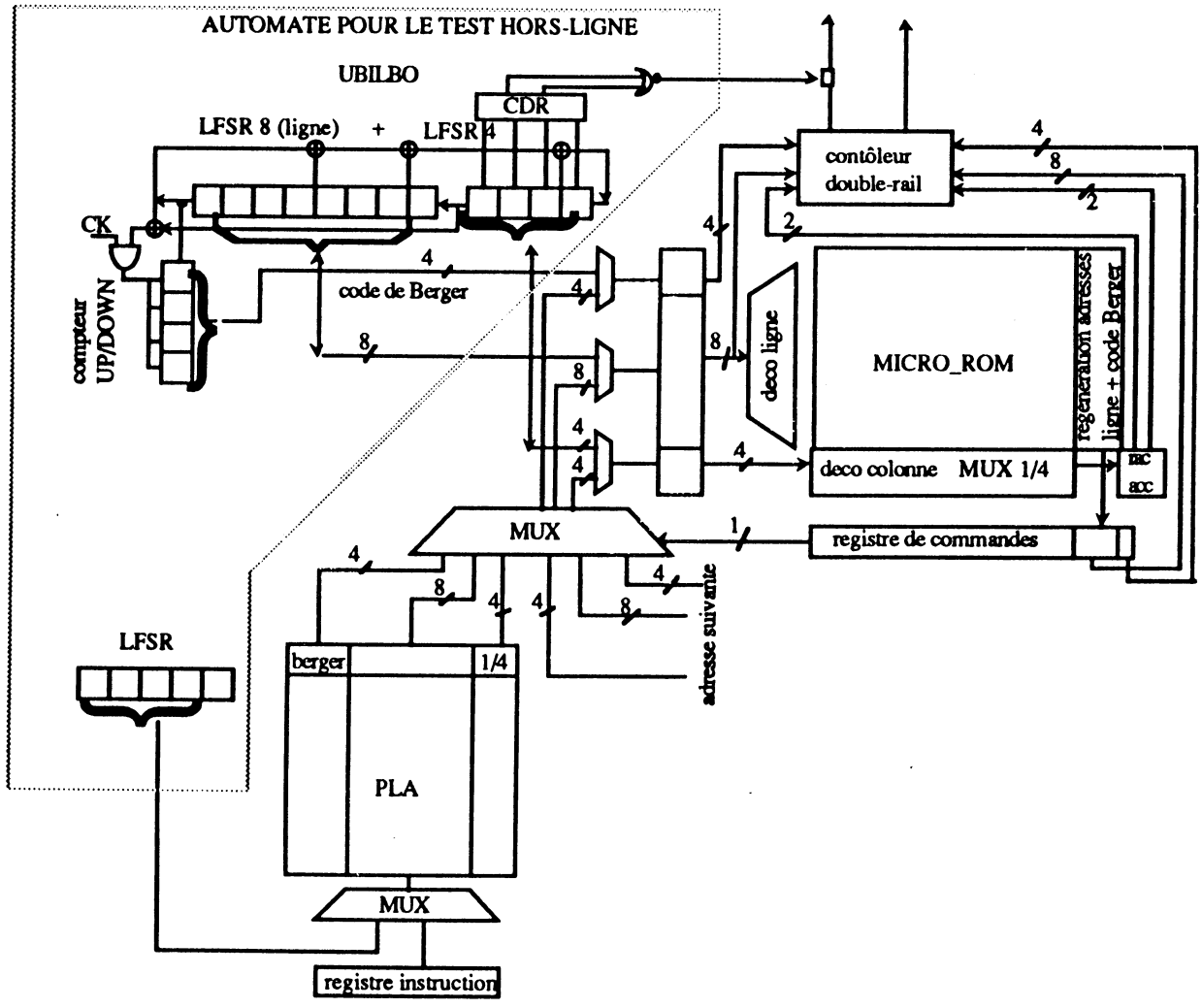


Schéma n°15 : Partie contrôle UBIST

Le multiplexeur des comptes-rendus reçoit tous les comptes-rendus au cours d'un cycle de 100 ms, on peut donc le considérer comme dynamisé.

Lors du test de la micro-ROM , toutes les adresses sont générées, cependant, certaines de ces adresses ne correspondent à aucune case mémoire et toutes les sorties se trouvent alors à 1. Ce problème peut être résolu de diverses manières :

- 1° On détermine exactement le nombre d'adresses non utilisables, puis on les répartit de telle sorte qu'un petit circuit de décodage puisse déterminer ces adresses. L'exemple suivant va illustrer cette méthode générale qui est décrite dans [NIC 90b].

Supposons que sur les 256 adresses lignes seulement 160 soient utilisées (160.4 = 640 ce qui correspond à l'ordre de grandeur du remplissage réel). Nous pouvons répartir les 96 adresses non utilisées de la manière suivante :

Il existe $2^6 = 64$ adresses dont les deux premiers bits sont à 1

Il existe $2^5 = 32$ adresses dont les trois premiers bits sont à 0

A partir de ces constatations, on peut concevoir le circuit du schéma n° 16 qui permet de générer un signal qui inverse une des sorties du contrôleur double-rail, qui contrôle la régénérescense des adresses dès que ces adresses sont présentes. On évite ainsi de détecter une erreur qui n'en est pas réellement une.

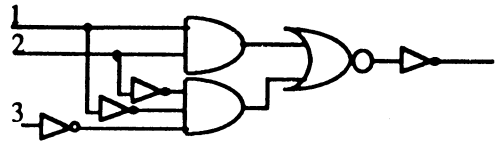


Schéma n°16 : Circuit de détection des adresses inexistantes

Un exemple dans lequel, le nombre d'adresses est impair donne un circuit plus complexe.

Si toutes les sorties de la micro-ROM sont à 1 alors l'adresse envoyée à la nano-ROM est hors-code. Une des sorties du contrôleur double-rail vérifiant les adresses devra donc être inversée. Dans ce cas, le décodeur colonne décode systématiquement les 4 colonnes puisque tous les bits d'adresse sont à 1.En conséquence, les mots présents dans chacun des deux registres de commande ne seront pas complémentaires, il faudra donc inverser aussi une des deux sorties du contrôleur double-rail.

Une seconde technique peut être envisagée.

Elle nécessite l'usage d'un circuit comparateur comme celui qui est décrit par le schéma n°17 mais étendu à 8 bits .

Les adresses de la ROM sont totalement remplies jusqu'à une adresse n qui est programmée dans le comparateur (supression de branches ou de transistors). Le comparateur donne donc un signal qui

inverse une des sorties du contrôleur double-rail, qui contrôle la régénérescence des adresses de la micro-ROM, chaque fois que l'adresse est supérieure à la valeur programmée.

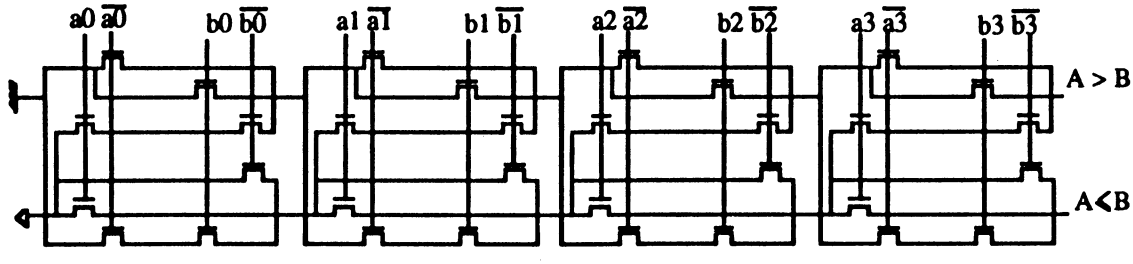


Schéma n°17 : comparateur 4 bits

Cette méthode à l'avantage d'être beaucoup plus souple du point de vue de la conception, mais elle est beaucoup plus lente à cause du nombre de transistors à traverser.

VII.3 L'ANALYSE DE SIGNATURE

Pour éviter d'avoir à utiliser un circuit complexe pour vérifier le circuit sous test, il est nécessaire de disposer d'une analyse de signature intégrée. De plus, il faut à la fois signaler les erreurs détectées par le contrôleur et par les analyseurs de signature au niveau du contrôleur double-rail final.

En règle générale, pour éviter de stocker une, ou plusieurs signatures de référence, on fait en sorte que l'analyseur de signature génère comme signature correcte finale, une séquence du type 1010101010..... . Cette séquence est associée à une bascule B qui fournit la séquence complémentaire 0101010101..... suivant le principe du schéma n°18 .

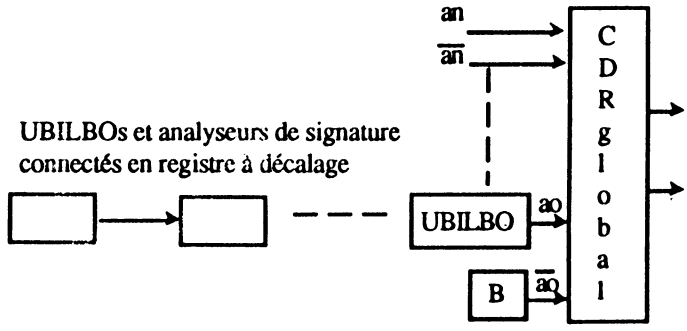


Schéma n°18 : Principe de vérification de la signature

Durant le test hors-ligne, on vérifie les différents blocs par comparaison avec leurs blocs duaux mais aussi par l'analyse de signature. L'analyse de signature a ici, pour rôle, de détecter les pannes qui

pourraient, par leurs effets complémentaires, rester indétectables par la comparaison. C'est pour cette raison que l'analyse de signature ne se fait que sur un seul bus et sur une des deux parties contrôle.
Les UBILBOs couvrent les pannes des PLAs, ils sont donc présents dans chacune des parties contrôle. Ces UBILBOs n'assurent pas un contrôle supplémentaire, comme dans le reste du circuit, mais le contrôle principal des PLAs.
Le circuit MAPS dispose de 4 analyseurs de signature :
- Les 2 UBILBOs qui contiennent la signature du test des PLAs.
- L'analyseur de signature d'un des deux BUS.
- L'analyseur de signature d'un des deux registres de commande issus d'une des deux PC.
- L'analyseur de signature des sorties

Ces analyseurs de signatures sont tous connectés en série (comme cela est décrit par le schéma n° 18) ce qui représente, pour le circuit MAPS, une longueur de 100, soit une couverture de l'ordre de 10^{-30} ou 2^{-100}.

VII.4 CONCLUSION

Grâce à l'intégration de générateurs de test spécifiques ou de logiciels appropriés, nous venons de réaliser un test exhaustif de presque tous les éléments du circuit et ceci toutes les 100ms. La technique UBIST est donc acquise. Ceci permet de couvrir toutes les pannes triples du circuit.

Cependant la très haute sécurité atteinte par le coeur du circuit ne doit pas être rendue inutile en utilisant des interfaces de médiocres qualités de sécurité .

Nous allons donc voir, par la suite, comment obtenir des interfaces aux qualités équivalentes à celles du circuit pour obtenir un système complet de très haute sécurité.

CHAPITRE VIII

INTERFACES STRONGLY FAIL-SAFE

VIII.1 INTRODUCTION

Le chapitre V nous a permis de donner une brève description du fonctionnement des entrées et des sorties. Ce présent chapitre va permettre d'en préciser le fonctionnement et les méthodes de test employées et ainsi nous permettre d'atteindre la propriété "strongly fail safe". Cette propriété, qui assure que le circuit est toujours capable de transmettre une donnée correcte ou sûre quel que soit son état interne, est bien entendue fondamentale pour la sécurité.

Ce chapitre sera divisé en deux parties, l'une décrivant le fonctionnement des entrées dites contrôlées et l'autre traitant des sorties au circuit. Ces deux parties suivront le même fil conducteur, à savoir l'obtention de la propriété "strongly fail-safe".

VIII.2 LES ENTREES CONTROLEES

VIII.2.1 : Obtention de la propriété fail-safe

La description qui a été faite au chapitre V schéma n°11 ne permet pas de garantir la présence d'entrées, soit correctes, soit sûres (Fail-safe). On peut remarquer que si une des sorties du LFSR est collée à 1 et que le contact est ouvert, il y a égalité des séquences car l'entrée est imposée à 1. Le résultat obtenu sera donc un état erroné dangereux.

Pour assurer la propriété fail-safe, on utilise un deuxième comparateur de séquence (dual du premier) par entrée (schéma n°1) ainsi qu'un deuxième LFSR.

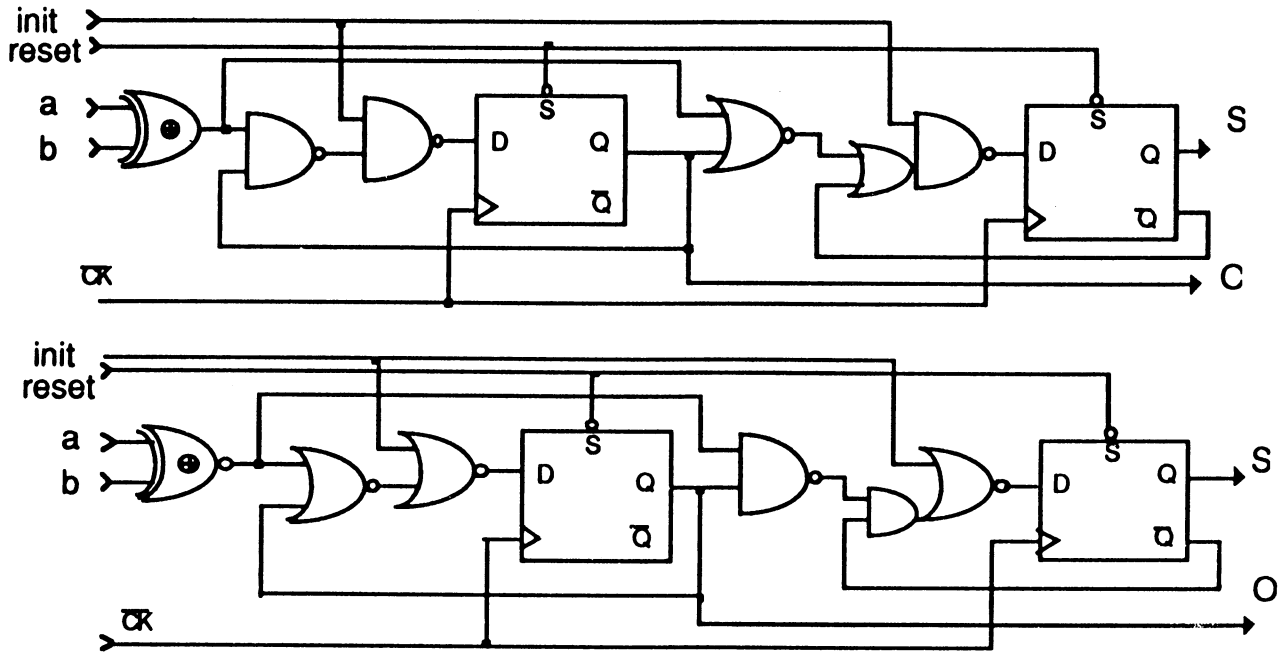


Schéma n°1 : Le comparateur de séquence et son dual

Le résultat des deux comparateurs, ainsi que le contenu d'un point intermédiaire O par comparateur, sont envoyés vers un contrôleur double-rail (voir schéma n°2). La sortie du comparateur est dirigée vers le registre d'entrée et la sortie du comparateur dual est dirigée vers le registre d'entrée complémentaire. Chacune des 8 entrées est aussi dirigée vers un générateur de parité, de même que les 8 entrées duales, de telle sorte que les données fournies aux bus contiennent 9 bits chacune.

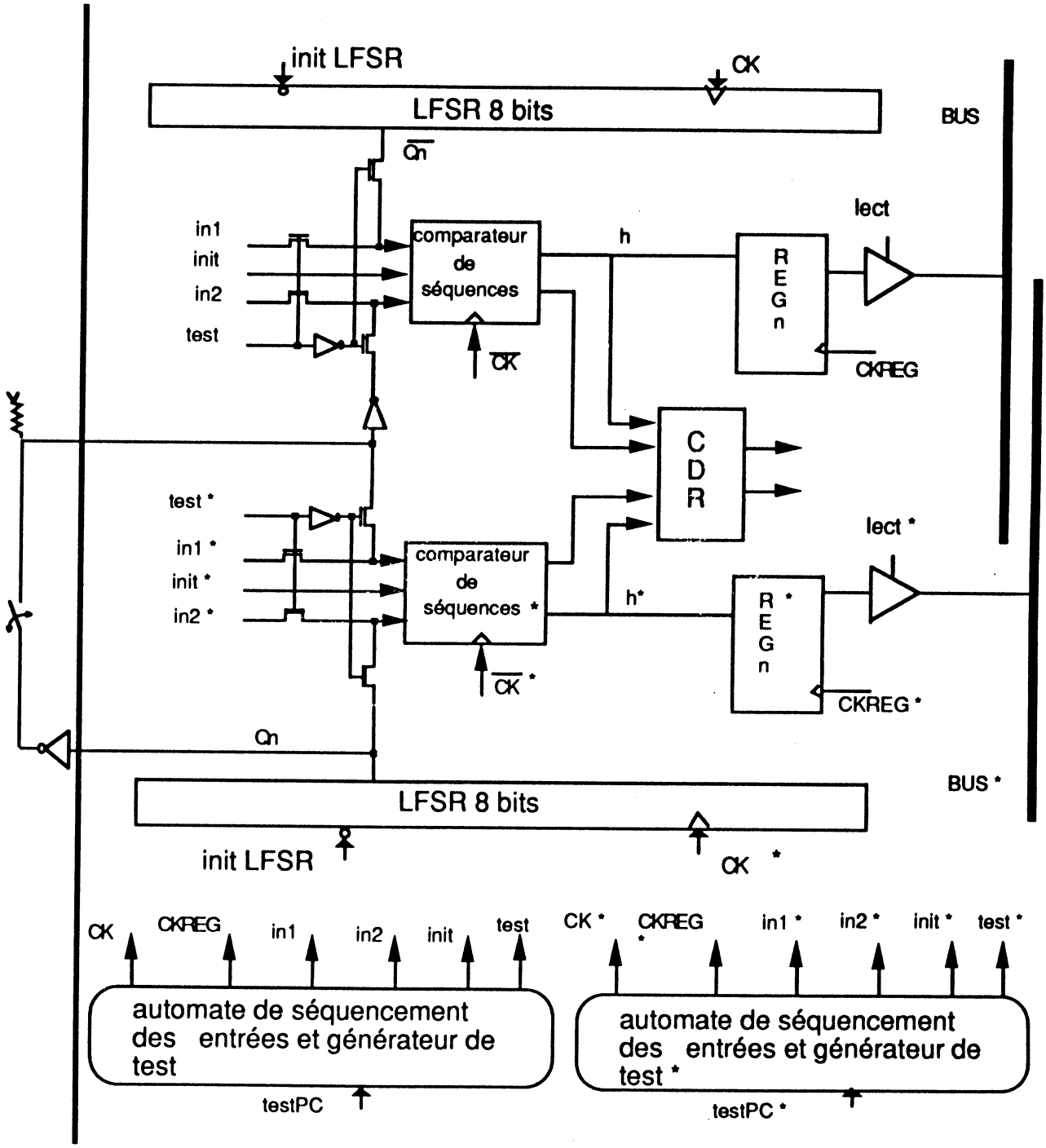


Schéma n°2 : Circuit de traitement des entrées

Du point de vue de la sécurité, on peut remarquer que le système est protégé contre toute panne n'affectant qu'un seul élément

VIII.2.2 : Obtention de la propriété strongly fail safe

Le dernier point délicat concerne les pannes, qui en se combinant pourraient inhiber la propriété "fail-safe". Pour éviter cette situation le circuit doit être activé régulièrement. On obtient ainsi la propriété "strongly fail-safe".

La preuve que cette propriété est bien assurée peut être faite en étudiant chacun des éléments qui composent cette interface. Le fait que le LFSR fonctionne en permanence, c'est à dire qu'il passe par tous ses états à chaque cycle, suffit à prouver que la propriété "strongly fail safe" est assurée pour les pannes affectant le LFSR.

Par contre, les comparateurs tolérant 1 bit d'erreur ne reçoivent pas systématiquement toutes leurs entrées et ne passent pas forcement par tous leurs états internes à chaque cycle. De plus, certaines entrées ne sont utilisées qu'exceptionnellement. Il faut donc que le circuit dispose d'un mécanisme propre capable d'activer complètement chacune des entrées du circuit.

Ce test est assuré par implantation d'un générateur de test adéquat qui envoie ses séquences (voir tableau n°1) aux 8 comparateurs de séquences, tandis qu'un second générateur de test envoie ses séquences aux 8 comparateurs de séquences duaux. La phase de test sera activée une fois par cycle lors du déroulement de l'instruction STOP.

testPC	0	1
ETAT	ETAT SUIVANT/in1,in2,init,test	ETAT SUIVANT/in1,in2,init,test
0000	0000 / X,X,X,0	0001 / 1,0,1,1
0001	0010 / 1,0,0,1	0010 / 1,0,0,1
0010	0011 / 0,0,0,1	0011 / 0,0,0,1
0011	0100 / 0,1,0,1	0100 / 0,1,0,1
0100	0101 / 1,1,0,1	0101 / 1,1,0,1
0101	0110 / 1,0,0,1	0110 / 1,0,0,1
0110	0111 / 1,1,0,1	0111 / 1,1,0,1
0111	1000 / 0,1,1,1	1000 / 0,1,1,1
1000	1001 / 0,0,1,1	1001 / 0,0,1,1
1001	1010 / 1,0,1,1	1010 / 1,0,1,1
1010	1011 / 0,1,1,1	1011 / 0,1,1,1
1011	1100 / 0,0,1,1	1100 / 0,0,1,1
1100	1101 / 1,1,0,1	1101 / 1,1,0,1
1101	1110 / 1,1,1,1	1110 / 1,1,1,1
1110	0000 / X,X,X,0	0000 / X,X,X,0

Tableau n°1 : Etats de l'automate de test associés aux séquences de test

Ces séquences ont été élaborées de telle sorte que chacune des portes du comparateur reçoive bien tous les couples de vecteurs qui permettent de la tester contre les pannes simples et les s-opens. Cette séquence assure aussi la présence de tous les états possibles générés par deux bascules.

Cependant, les séquences de test ne permettent pas de générer des mots hors-code qui assureraient la propriété "strongly code disjoint", le contrôleur double-rail n'est donc testé que pour toutes les pannes simples et la majorité des pannes multiples [NAN 88]. Le test est déclenché par la partie contrôle grâce au signal testPC, à partir de cet instant l'automate génère les 14 séquences de test puis envoie un compte-rendu de fin de test à la partie contrôle qui peut alors commander un autre test. La phase de test est effectuée à une fréquence beaucoup plus élevée (1 MHz) que la fréquence d'acquisition des 32 bits de la séquence échangée, d'où le multiplexage des fréquences d'horloge présenté par le schéma n°3 .

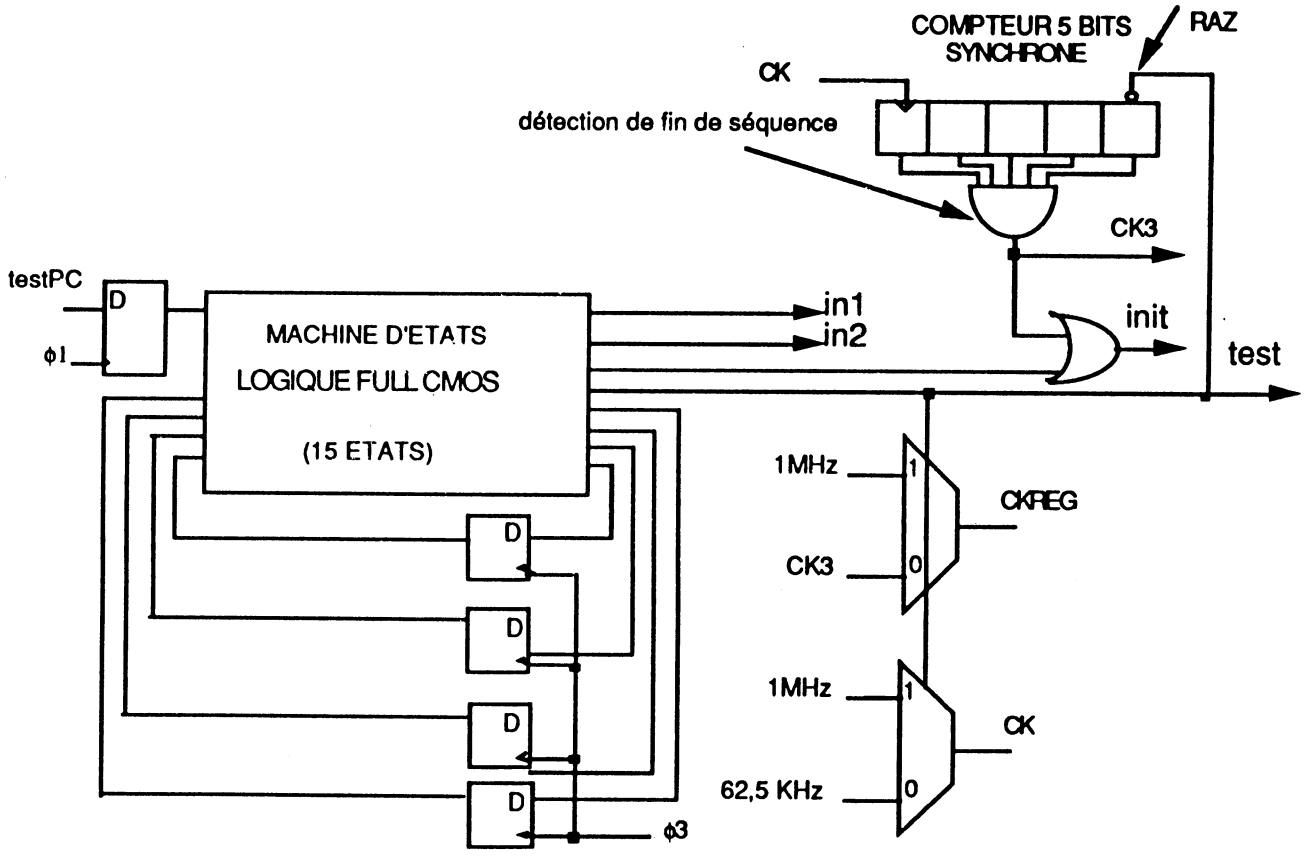


Schéma n°3 : Fonctionnement général de l'automate

Le détail de la machine d'état de 15 états est présenté par le schéma n°4 . Le choix d'utiliser des bascules D et des portes logiques pour concevoir l'automate repose sur le fait que la machine ne dispose que d'un très faible nombre d'état, ce qui donne une surface beaucoup plus faible que si on avait utilisé un PLA.

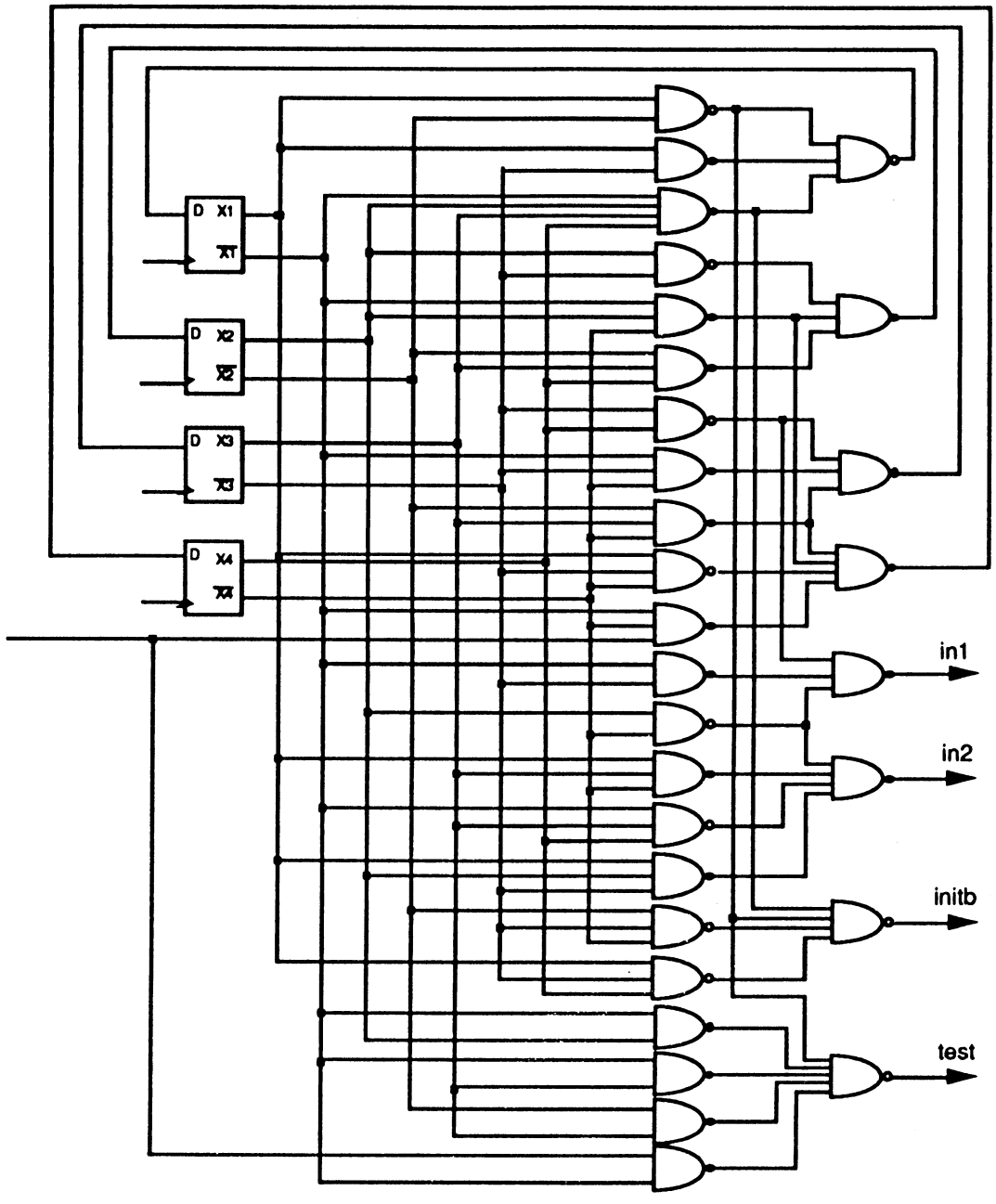


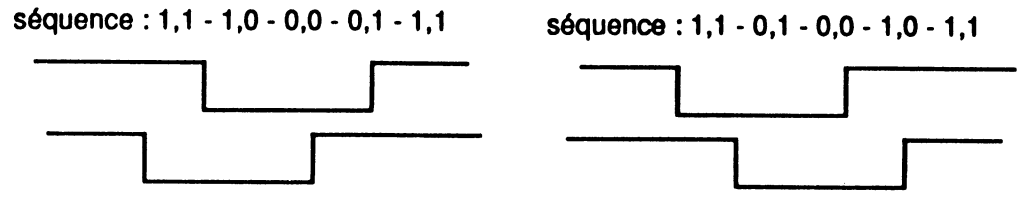
Schéma n°4 : Automate de test

3 : L'automate de comptage d'essieux

Les entrées peuvent aussi avoir la fonction particulière de compter les essieux et de donner le sens des trains qui circulent sur la voie. Seules 2 entrées sont réservées à cet usage.

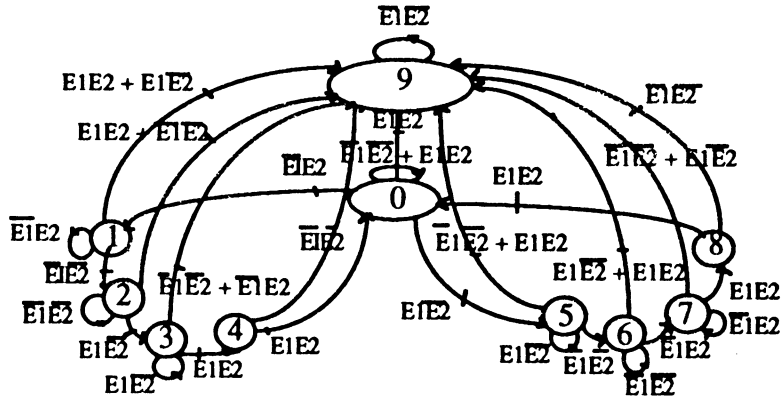
Cette fonction supplémentaire nous oblige à concevoir un second automate. Son rôle est de scruter le résultat de la lecture des entrées pédales et de réaliser les fonctions demandées.

Le comptage des essieux se fait modulo 512 (norme SNCF). Pour reconnaître le sens d'un train, il faut pouvoir distinguer deux séquences sur les signaux provenant des entrées pédales à savoir :



A chaque fois qu'une de ces séquences est détectée, le signal de sens du train est calculé et mis à jour et le comptage est incrémenté. Le comptage exacte nous importe peu puisqu'il sagit d'effectuer un comptage comparatif à l'aide de deux MAPS placés en deux points de la voie (entrée et sortie d'une gare).

La machine d'états finis peut être décrite par le graphe du schéma n°5.



ETATS 0,1, 2, 3, 5, 6, 7 --> SORTIES (D, G, I) = (0, 0, 0)

ETAT 4 --> SORTIES (D, G, I) = (1, 0, 1)

ETAT 8 --> SORTIES (D, G, I) = (0, 1, 1)

ETAT 9 --> SORTIES (D, G, I) = (1, 1, 0)

Schéma n° 5: Graphe d'états du compteur d'essieux

L'automate correspondant à ce graphe possède deux entrées E1 et E2 et trois sorties G (gauche), D (droite) et I (incrémenter). Il sera donc implémenté à l'aide de 4 bascules et plusieurs portes logiques, sa complexité est du même ordre que l'automate précédemment décrit. La sortie I est branchée sur l'horloge d'un compteur de 9 bits, ce compteur est incrémenté chaque fois que I = 1 et quel que soit le sens du train puisque l'on comparera deux comptages.

L'état 9 est l'état dans lequel se retouve le système si une transition (E1, E2) ne correspondant pas à la succession décrite par le schéma n° 5. (E1, E2) = (1, 1) permet de revenir directement à l'état 0 , ainsi au moins un essieu ne sera pas compté. Pour augmenter la sécurité de cet automate il suffit d'augmenter le nombres d'états entre 9 et 0 et ainsi être sûr que 2 ou 3 ou 4 essieux ne seront pas comptés. Etant donné que les automates sont dupliqués, si les comptage sont différents, le circuit

MAPS pourra dire si l'un des deux automates est en panne (indication d'erreur global) , si ce sont les pédales qui sont défectueuses (ou s'il y perte d'un essieu) la comparaison des comptages entre deux MAPS différents permettra de le détecter.

Le résultat de ce comptage doit être lu à chaque cycle afin qu'il soit envoyé par message codé au poste d'aiguillage . Pour cela, il est gardé dans deux registres dont la lecture est commandée par la PC. La lecture permet de faire transiter le contenu de ces registres, via le BUS dans des octets réservés de la RAM . Le comptage et le sens nécessitent deux octets. Le passage par le bus demande l'ajout d'un bit de parité à chacune des données. La gestion de cette parité, ainsi que celle de l'octet d'information, est présentée par le schéma n° 6.

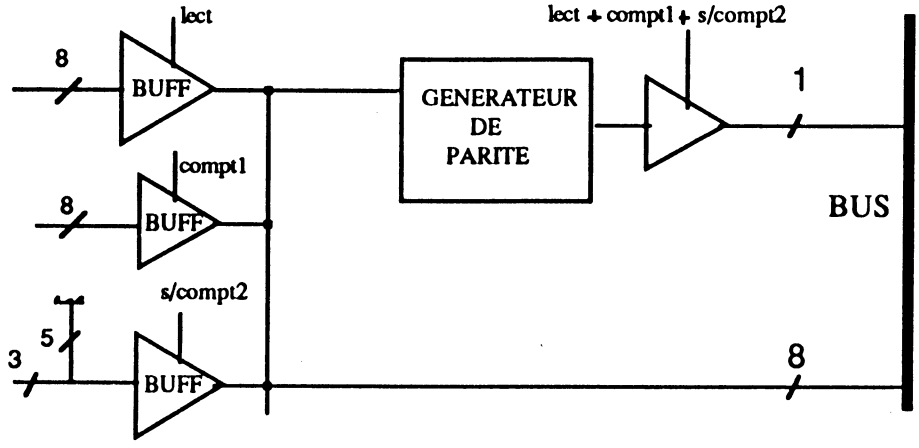


Schéma n°6 : Gestion des parités et des données avec le bus

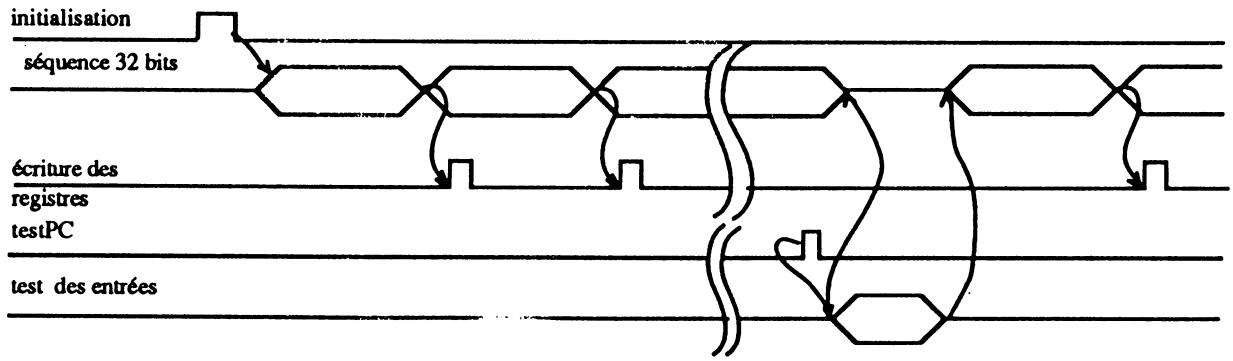


Schéma n°7 : Séquencement des entrées

Le schéma n°7 présente un diagramme simplifié du séquencement du circuit de traitement des entrées. Le schéma n°8 donne une description mi-structurelle et mi-topologique qui pourra être la première ébauche d'un plan de masse des entrées. Le véritable plan de masse devra prendre en compte la taille réelle de chacun des blocs.

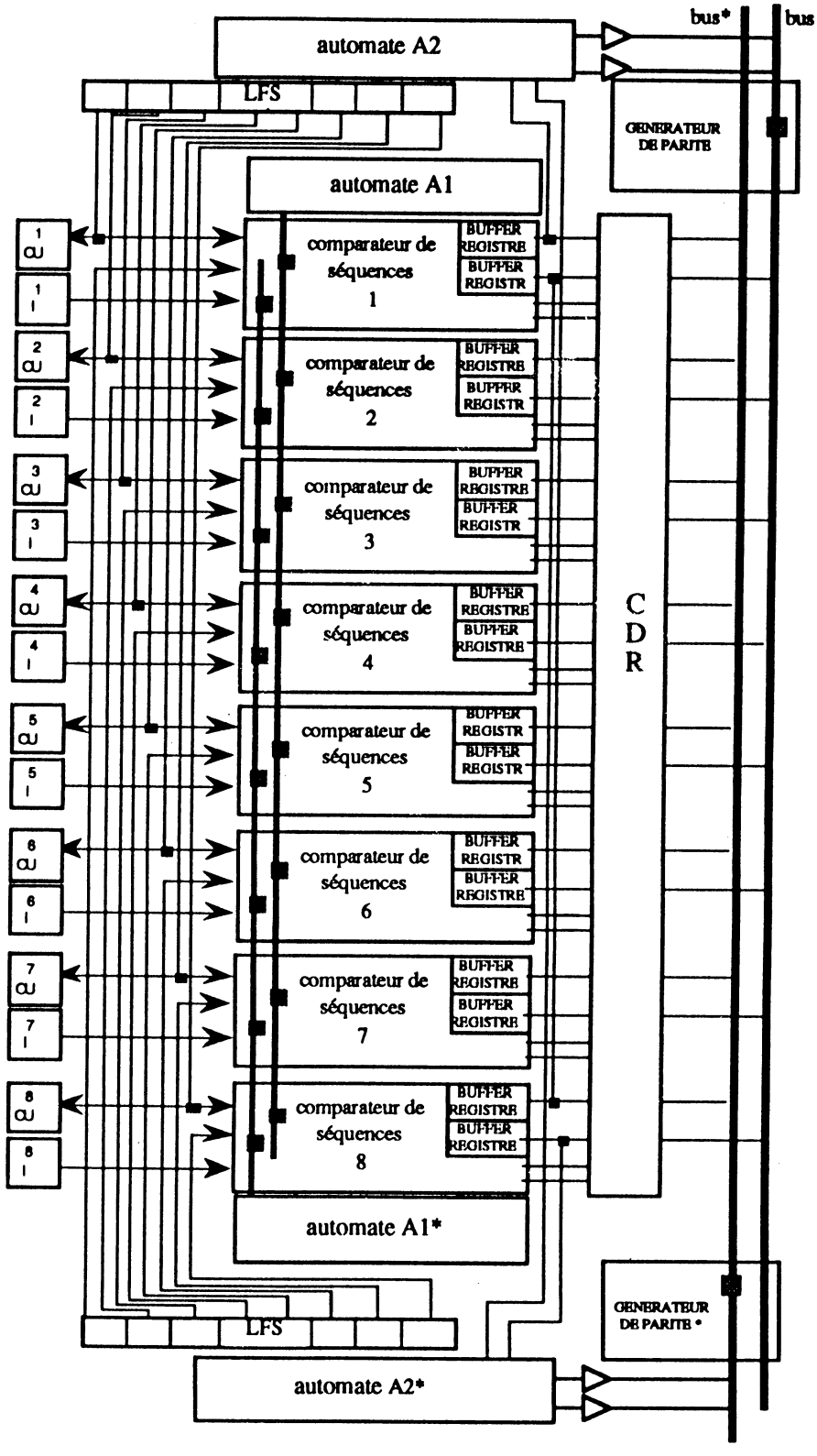


Schéma n° 8 : Aspect topologique des entrées

VIII.3 LES SORTIES FREQUENCES

VIII.3.1 : Interrupteur fail-safe

La réalisation de l'interrupteur f"ail-safe" est basée sur celui qui a été proposé dans [NIC 89], il est décrit par le schéma n°9 .

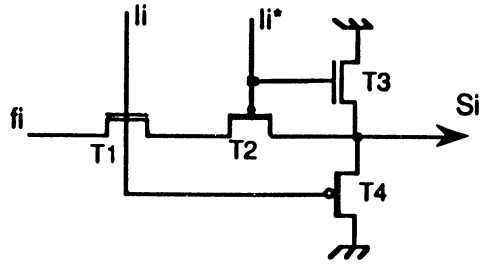


Schéma n°9 : interrupteur fail-safe

Les informations, codées en double-rail, générées par la partie opérative et la partie opérative duale, sont utilisées pour bloquer ou pour laisser passer les fréquences fi sur les sorties Si.
Dans cette figure, on voit que la fréquence fi est transmise sur la sortie Si si Ii = 1 et Ii* = 0, ceci rend passant les transistors T1 et T2 et bloque les transistors T3 et T4. On peut facilement vérifier qu'aucune panne simple ne pourra connecter fi sur Si tant que Ii = 0 et Ii* = 1 (excepté un court-circuit entre fi et Si que l'on rendra impossible en écartant topologiquement fi et Si). On peut aussi vérifier que l'on est également protégé contre les défaillances de la partie opérative car la génération sur Ii et Ii* de valeurs non codées en double-rail (e.g. Ii = 1, Ii* = 1 ou Ii = 0, Ii* = 0) déconnecte la sortie Si de la fréquence fi. Néanmoins cette protection n'est pas nécessaire car les défaillances de la partie opérative sont détectées par des contrôleurs appropriés qui agissent par ailleurs sur un mécanisme de coupure de l'alimentation du circuit que l'on verra par la suite. Notons aussi que les transistors T3 et T4 ne sont pas vraiment nécessaires mais leur présence permet d'éviter la mise de Si en haute indépendance chaque fois que Si est déconnectée de fi, ceci réduit la sensibilité de Si vis à vis des perturbations externes. De plus, en dimensionnant les transistors de façon adéquate (i.e. transistors T3 et T4 très larges), on peut augmenter la sûreté du système qui sera cette fois "fail-safe" pour des pannes doubles (e.g. T1 et T2 stuck-on, I1 stuck-at 1 et T2 stuck-on etc...). La raison est la suivante : les transistors T3 et T4 étant peu résistifs, si l'un ou les deux sont passants, la valeur 0 sera imposée sur la sortie Si même si celle-ci est connectée à fi.
Une simulation électrique de ce circuit [CAS 89] a permis de valider cette architecture.

Si l'on voulait augmenter la sécurité, on pourrait augmenter le nombre des transistors séparant la sortie Si de la source de fréquence fi en ajoutant deux transistors T'1 et T'2 . Dans ce cas, la sortie Si ne prend pas de façon erronée la valeur 1 (présence de la fréquence fi) même si trois des transistors

T1, T2, T1', T2' restent passants. Bien sûr, la sécurité sera plus élevée si l'on augmente la largeur des transistors T3 et T4.

L'interface complète correspondant aux 8 sorties monorail dynamisées est présentée dans le schéma n° 10. Dans cette figure on voit aussi le placement du contrôleur double rail qui vérifie l'intégrité des données (i.e. I1, I1*, I2, I2* ... I8, I8*) générées par la partie opérative.

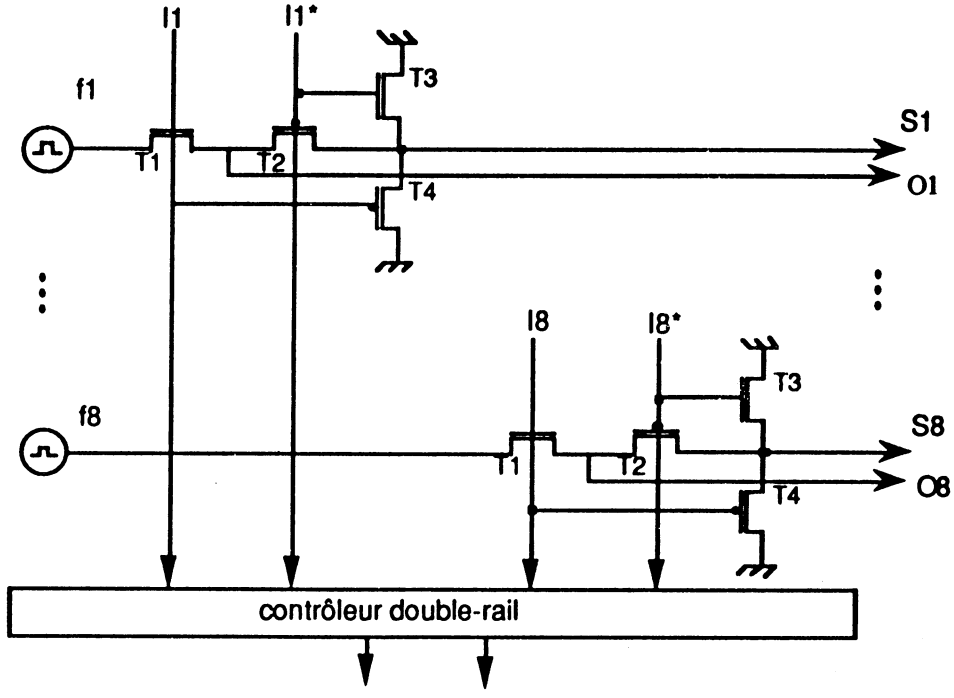


Schéma n°10 : Interface de sortie

VIII.3.1 : L'interface "Strongly fail-safe"

D'après les explications de la section précédente, si une panne survient dans l'interface du schéma n°10 , la sortie Si ne prend jamais la valeur 1 de façon erronée et la sécurité du système est assurée. Néanmoins ce schéma n'a aucun mécanisme de détection des pannes, ainsi si une panne survient dans l'interface elle reste indétectable et après un temps suffisamment long, une 2ème panne peut survenir. De cette façon plusieurs pannes peuvent s'accumuler et pour la combinaison de certaines pannes la sortie Si peut prendre de façon erronée la valeur 1 (présence de la fréquence fi). Un tel cas survient par exemple lors de la présence simultanée des pannes : T1 s-on, T2 s-on, T3 s-open et T4 s-open. Les pannes conduisant à une telle situation, doivent être détectées pour assurer la propriété "strongly fail-safe". Leur détection va déclencher la coupure de l'alimentation du circuit par le mécanisme présenté dans la section suivante. La détection des pannes sera assurée par la génération de vecteurs de test et un analyseur de signature (voir schéma 11).

Deux signaux (TEST et TESTB) sont utilisés pour reconfigurer le circuit en mode de test, ainsi l'activation erronée d'un de ces signaux (TEST = 1 ou TESTB = 0) va affecter uniquement les valeurs des I1,... , I8 ou uniquement les valeurs complémentaires I1*,... , I8*, la détection d'une telle situation sera effectuée par le contrôleur double-rail. Dans le cas où "TEST" est activé (TEST = 1) de façon erronée, il y a isolation des entrées fréquence de l'interface. L'activation erronée du TESTB (TESTB = 0) connecte VDD aux entrées de l'interface, dans ce cas un large dimensionnement des transistors T1, T2,... ,T8 permettra d'imposer la valeur électrique de VDD aux entrées fréquence de l'interface. Par conséquent on est doublement protégé contre l'activation erronée du TEST ou du TESTB (i.e. détection par le contrôleur double-rail et génération aux sorties de l'interface des valeurs sûres c.a.d différentes des fréquences fi).

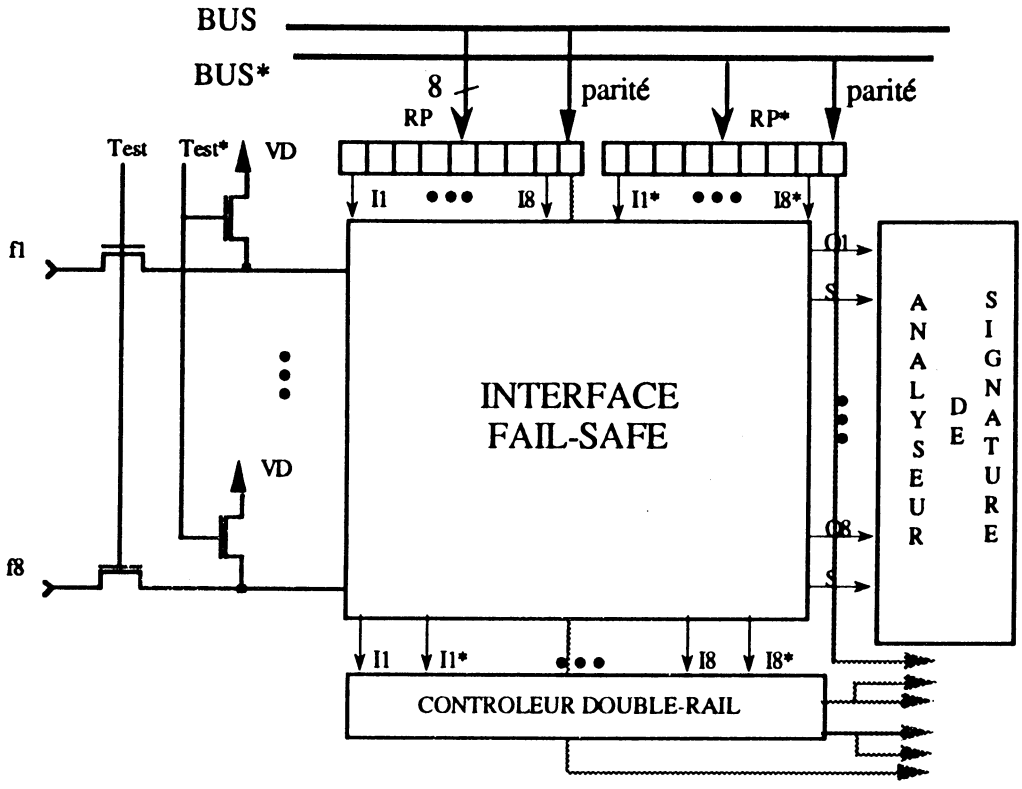


Schéma n° 11 : Interface "Strongly fail-safe".

Comme cela a déjà été mentionné, pour avoir une interface "strongly fail-safe", certaines pannes qui peuvent amener par leur accumulation à une situation dangereuse, doivent être détectées. En se référant au schéma n° 10, on peut constater que ces pannes sont : T1 s-on, T2 s-on, T3 s-open, T4 s-open, Ii stuck-at 1, Ii* stuck-at 0, ainsi que certaines pannes équivalentes aux précédentes, telles qu'un stuck-at 1 sur les grilles des T1 et T2, ou un stuck-at 0 sur les grilles des T3 et T4, ainsi que les courts-circuits entre source et drain de T1 et T2, ou les coupures des équipotentielles connectées

sur la source ou le drain de T3 ou T4 etc... En considérant comme point d'observation (entrées de l'analyseur de signature) les points 0i et Si (voir le schéma n°9), le tableau n°2 donne la séquence de test à appliquer sur Ii et Ii*, les valeurs respectives sur 0i et Si et les pannes détectées. Les autres pannes qui ne sont pas présentées sur le tableau n°2 sont détectées par équivalence. Du point de vue fonctionnel, cette séquence permet de vérifier :

- si le point 0i est isolé de la fréquence fi et si la sortie Si est connectée à la masse, chaque fois que Ii est égal à 0.

- si la sortie Si est isolée du point 0i et si elle est connectée à la masse, chaque fois que Ii* est égal à 1.

Le signal CE indique la présence d'un mot du code ou d'un mot hors-code.

Ii	Ii*	Oi	Si	Pannes détectées	CE
1	0	1	1		0
0	1	1	0	T2 s-on, Ii* stuck-at 0	0
1	0	1	1		0
0	0	0	0	T3 s-open	1
0	1	0	0	T1 s-on, I1 stuck-at 1	0
1	0	1	1		0
1	1	1	0	T4 s-open	1

Tableau n°2 : séquences de test.

Le contrôleur double-rail du schéma n°10 est conçu de la même façon que celui du bus et il est testé en même temps que lui grâce au générateur de test présenté par le schéma n°2 chapitre VII. Certains de ces vecteurs testent aussi l'interrupteur fail-safe, il reste cependant 4 vecteurs à générer pour assurer la présence de tous les vecteurs du tableau n°2 en commande de l'interrupteur. Ces vecteurs sont générés à partir de l'UA par microprogramme. La séquence de vecteur est la suivante :

registre de sortie	registre de sortie dual
11111111	00000000
00000000	11111111
11111111	00000000
11111111	11111111

Il est facile de remarquer que cette séquence de vecteurs n'est pas présente dans le tableau n° 3 du chapitre VII . Nous avons vu au chapitre VII schéma n°8 comment était modifié, grâce à des

commandes de la PC, l'automate générant les signaux d'indication d'erreur pour indiquer que le 4ième vecteur est hors-code.

Il faut cependant remarquer que l'état des sorties doit être conservé pendant le test et que ce temps doit être suffisamment court pour ne pas perturber l'état des sorties.
Pour résoudre le premier problème, il suffit, avant de débuter le test des contrôleurs double-rail du bus et des sorties, de mémoriser le contenu des registres de sortie dans les RAMs. Dès que le test est terminé et que les 4 séquences décrites ci-dessus ont été envoyées, on restaure l'état des registres de sortie. Ce temps de test dure environ 40 μs, ce qui est très inférieur au temps de réaction des circuits externes. Le test est donc vu comme une courte perturbation des signaux de sortie qui est totalement éliminée par les filtres passe bande externes.

Le dernier point concerne la longueur de l'analyseur de signature. Cette longueur peut être augmentée à volonté afin d'assurer un masquage moyen très faible (masquage moyen égal à 2^{-n} avec n la longueur de la signature). Une autre possibilité consiste à simuler l'interface pour les différentes pannes (une telle simulation est réaliste à cause de la faible taille de l'interface présentée) et de vérifier s'il y a masquage, si cela est le cas, on peut modifier la longueur jusqu'au moment où la simulation donne une couverture de panne de 100 %.

Le circuit MAPS génère 8 sorties fréquence qui commandent des actionneurs situés sur les voies, il génère aussi une 9ième fréquence qui commande directement l'alimentation du circuit, l'absence de cette fréquence indique la présence d'une erreur dans le circuit. L'interrupteur est identique au 8 autres mais ses commandes sont issues de la mémorisation d'erreur et non plus de la partie opérative (voir schéma n° 12).

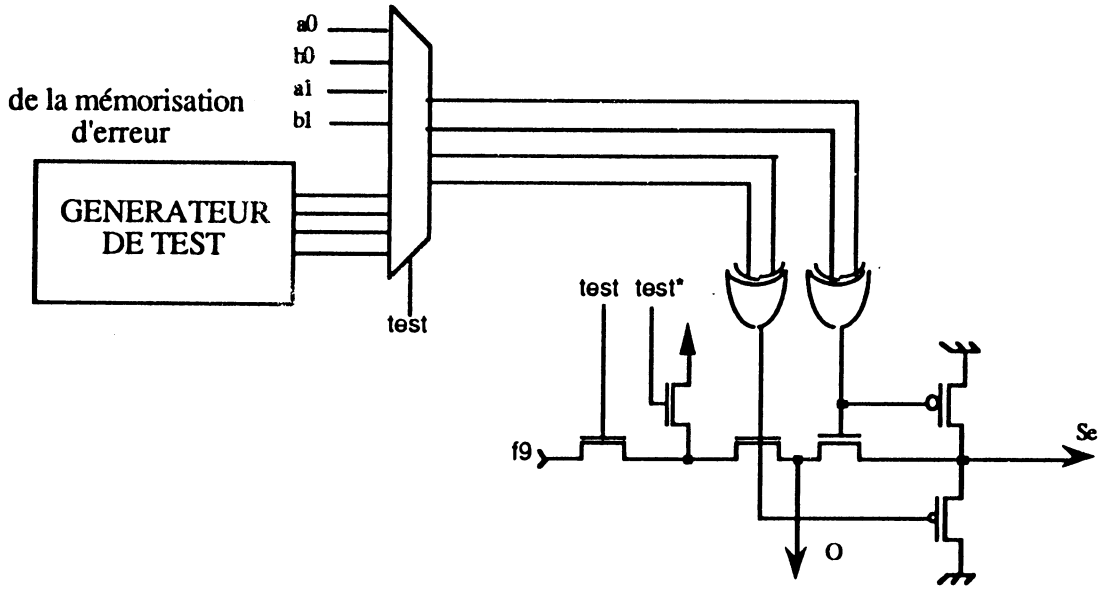


Schéma n°12 : Sortie S9

Cette sortie, qui se différencie des 8 autres, est traitée séparément. Cet interrupteur doit bien évidemment être "strongly fail-safe". Il faut donc qu'il puisse recevoir, comme pour les 8 autres sorties, la séquence de test présentée par le tableau n°2, mais il faut aussi qu'il reçoive des vecteurs capables de tester les portes XOR. Cette nouvelle architecture nous oblige à utiliser un générateur de test qui génère la séquence présentée par le tableau n°3 . Le circuit est reconfiguré en mode test grâce aux signaux test et test*. Les signaux Oe et Se sont injectés dans le même analyseur de signature que les 8 autres sorties.

Etant donné la similitude du circuit du schéma n° 12 à celui du schéma n°10, la séquence de test est basée sur la séquence de test du tableau n° 2, mais comme les transistors P (et N) sont placés différemment, le tableau n° 2 est modifié et on obtient la séquence II du tableau n°3 à appliquer sur A et B, ainsi que les valeurs attendues sur 0e, Se. La séquence à appliquer sur a0, a1, b0, b1, en prenant en compte les pannes sur les portes XOR, est donnée par la séquence III du tableau n°3. Elle est obtenue en répétant 2 fois la séquence II. La première fois on a remplacé A = 1 par [a0, a1] = [0,1], A = 0 par [a0, a1] = [0,0], B = 1 par [b0, b1] = [0,1], B = 0 par [b0, b1] = [0,0]. La deuxième fois on a remplacé A = 1 par [a0, a1] = [1,0], A = 0 par [a0, a1] = [1,1], B = 1 par [b0, b1] = [1,0], B = 0 par [b0, b1] = [1,1]. Ceci permet de vérifier si le test du tableau II se déroule aussi bien quand la valeur de A = 1 est obtenue par les valeurs [a0, a1] = [0,1] ou par les valeurs [a0, a1] = [1,0]. La même vérification est faite quand la valeur A = 0 est obtenue par les valeurs [a0, a1] = [0,0] ou [a0, a1] = [1,1]. Ceci est aussi valable pour les différentes valeurs de B, b0, b1.

Ce générateur de test est implémenté sous forme d'un automate spécifique comportant 4 bascules.

A	B	Oe	Se
1	1	1	1
0	0	1	0
1	1	1	1
0	1	0	0
0	0	0	0
1	1	1	1
1	0	1	0

Table II

ao	a1	bo	b1	Oe	Se
0	1	0	1	1	1
0	0	0	0	1	0
0	1	0	1	1	1
0	0	0	1	0	0
0	0	0	0	0	0
0	1	0	1	1	1
0	1	0	0	1	0
1	0	1	0	1	1
1	1	1	1	1	0
1	0	1	0	1	1
1	1	1	0	0	0
1	1	1	1	0	0
1	0	1	0	1	1
1	0	1	1	1	0

Table II I

Tableau n°3 : Séquence de test de la sortie S9

Finalement, la durée de test est de 14 cycles de l'horloge de base du circuit soit 14 µs. Ce temps doit être au maximum égale à 10% de la période de la fréquence de sortie pour ne pas perturber le signal de sortie, c'est à dire que :

$1{,}4.10^{5} < 10^{-1} T \Rightarrow f < \frac{1}{1{,}4} 10^{4} = 7140\ Hz$

On choisira donc une fréquence f9 inférieure à cette valeur.

Remarque : La sécurité peut aussi être atteinte en n'utilisant alternativement pour chaque sortie que deux fréquences, mais en prenant soin d'assurer un éloignement topologique suffisant pour éliminer les couplages entre deux même fréquences. Dans notre cas le cahier des charges nous a imposé 9 fréquences différentes, cette solution augmente sensiblement le nombre de transistors des sorties (compteurs, décompt...).

VIII.4 CONFIRMATION D'ERREUR

Le principe général des sorties nous permet de constater qu'en cas de problème (apparition et mémorisation des couples (0,0) ou (1,1) aux sorties du contrôleur global) le système bloque immédiatement la sortie S9, c'est à dire l'alimentation du circuit. Le circuit externe d'alimentation est présenté par le schéma n°13 .

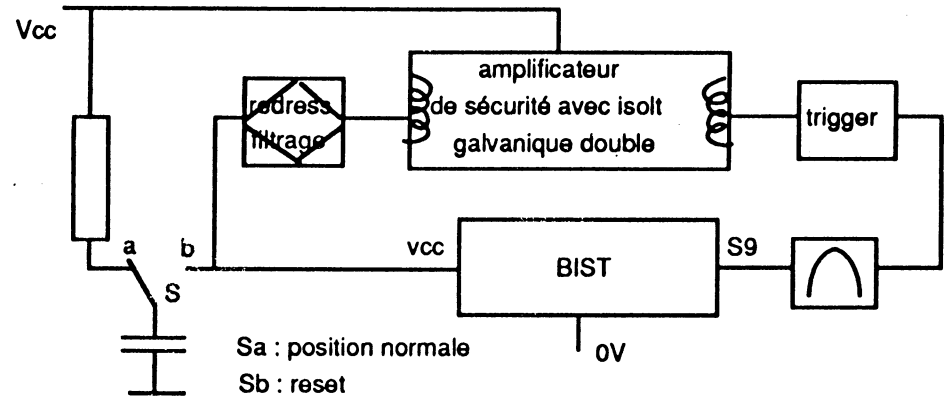


Schéma n°13 : Contrôle externe de l'alimentation

Faute d'alimentation, le circuit pert son contexte (tous les registres sont à 0). Cet état est signalé au poste d'aiguillage où un opérateur peut reprendre le réseau en manuel pour faire évoluer un convoi vers un lieu connu. Une fois cette opération terminée, l'opérateur va effectuer un test de confirmation.

Pour cela, il alimente le circuit à l'aide d'un condensateur qui effectue un"reset" du circuit. Ce reset réarme les bascules de mémorisation (f9 est donc restaurée) et peut aussi imposer des valeurs complémentaires dans certains registres (l'accumulateur, les entrées, les sorties) de la partie

complémentaire du circuit. Ce reset doit aussi forcer le registre d'entrée de la micro-ROM à une adresse d'exception qui serait celle du début du programme d'initialisation. Le programme d'initialisation assure que tous les registres complémentaires reçoivent bien des valeurs complémentaires (principalement les RAMs) . Une fois l'initialisation terminée, le programme se branche automatiquement au début de l'instruction STOP pour effectuer un test, c'est le test de confirmation.

Ce test de confirmation est identique au test hors-ligne périodique mais il ne se pratique que lors de la mise sous tension du circuit ou lors de la vérification du circuit après détection d'une erreur, c'est à dire lorsque le circuit est , ou a été vidé.

Si une nouvelle erreur se produit, la fréquence f9 est à nouveau coupée et le circuit effectue des opérations jusqu'à épuisement de sa charge, le circuit devra être changé. Si le test de confirmation dément l'erreur, f9 est maintenue et le circuit peut reprendre le programme de traitement au début.

Le circuit ne peut donc en aucun cas prendre l'initiative de son réarmement car cela ne peut être fait que si on connait exactement l'état du réseau.

Vu que les pannes transitoires sont beaucoup plus fréquentes que les pannes permanentes, il nous a semblé intéressant de disposer d'une télécommande pour effectuer le reset de confirmation.

VIII.5 CONCLUSION

La description des interfaces d'entrée et de sortie qui vient d'être faite nous permet d'intégrer complètement, et pour la première fois, des interfaces "strongly fail-safe".

Nous avons donc détaillé, au cours de ce chapitre, quelles étaient les limitations des interfaces sûres ou correctes (fail-safe) et les modifications qu'il fallait leur apporter pour qu'elles deviennent "strongly fail-safe". Ces modifications sont principalement dues à l'ajoût de générateurs de test spécifiques, capables de générer tous les vecteurs détectant les pannes, dont l'accumulation entraînerait la perte de la propriété "fail-safe".

On peut aussi remarquer que le coût et l'encombrement de ces interfaces leur permettent de remplacer avantageusement les systèmes actuels à base de relais. De plus, l'utilisation en sortie de fréquences, permet à cette interface de disposer d'un champ d'application assez étendu puisque de nombreux actionneurs sont commandés ainsi.

CHAPITRE IX

EVALUATION

IX.1 INTRODUCTION

Ce chapitre, basé sur des évaluations théoriques, doit nous permettre de disposer d'un ordre de grandeur des constantes fondamentales d'un circuit de sécurité, à savoir, le nombre de sorties, le nombre de transistors et l'évaluation de la fiabilité du circuit.

Ce chapitre est avant tout une transition nécéssaire entre la description fonctionnelle, présentée dans les chapitres précédents, et le début du développement correspondant au layout. Le layout n'est envisageable que si les chiffres théoriques acquis (avec la réserve qu'ils méritent) prévoient l'obtention d'un produit final conforme aux résultats attendus.

Nous débuterons ce chapitre par une première partie relative à l'estimation du nombre de transistors, en passant en revue chaque bloc successivement.

La seconde partie sera réservée au calcul de la fiabilité prévisionnelle du circuit. Ce calcul s'appuie sur les données numériques et les courbes fournies par le CNET (Centre National d'Etude et de Telecommunication).

IX.2 ESTIMATION DU NOMBRE DE TRANSISTORS

Le micro-contrôleur MAPS est un circuit auto-contrôlable obtenu grâce à l'utilisation de techniques récentes, il sera donc du type full-custom. L'estimation du nombre de transistors regroupe les transistors de tous les blocs décrits par le schéma n° 1. Ce schéma de synthèse correspond à la fusion du schéma n° 1 du chapitre VII et des deux interfaces décrites au chapitre VIII. La RAM double-accès est la même que celle décrite par le schéma n° 6 du chapitre V car les données qu'elle véhicule ne sont protégées que par le codage, aucun matériel supplémentaire n'est donc rajouté. La ROM externe est un produit commercial.

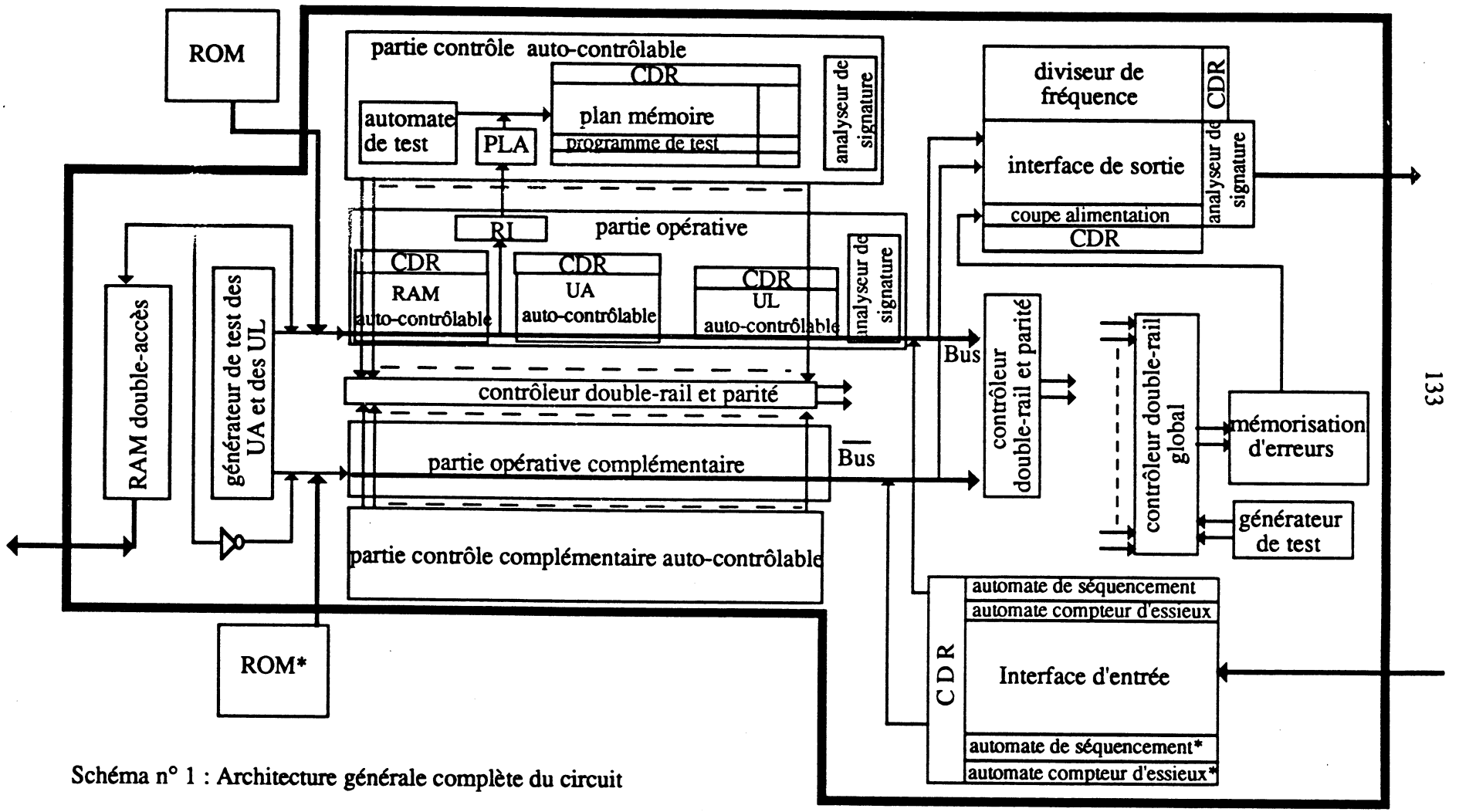


Schéma n° 1 : Architecture générale complète du circuit

IX.2.1 : Partie opérative

élément	nbr transistors	nbr éléments par tranche	nbr bits par tranche	multiplicité	sous-total
RAM interne	10	64	9	2	11520
UA	50	1	8	2	800
accumulateur A	22	1	8	2	352
registre B	14	1	8	2	224
UI + registre C	88	1	1	2	176
générateur de parité	55	1	1	6	330
RI	8	1	8	2	128
RAM double-accès	10	62	8	1	4960
multiplexeurs	50	1	1	2	100
générateur de test	500	1	1	1	500
CDR bus + CNCI	250	1	1	1	250
CDR annexes	250	1	1	2	500
analyseur signature	160		1	1	160
TOTAL					20000

Tableau n° 1 : Nombre de transistors de la partie opérative

IX.2.2 : Partie contrôle

	nbr bits/mot	nbr mots	nbr transistors/bit	multiplicité	sous-total
micro-ROM	34	560	1	2	38080
nano-ROM	70	220	1	2	30800
TOTAL					68880

Tableau n° 2 : Somme des transistors de ROM

	nbr de transistors	multiplicité	sous total
PLA	250	2	500
automate test + CDR	500	2	1000
MUX sélection	128	2	256
décodeurs	1000	2	2000
CDR commandes	222	1	222
analyseur signature	1100	1	1100
TOTAL			5078

Tableau n° 3 : Nombre de transistors hors-micro-ROM

La partie contrôle et sa duale nécessitent donc un nombre approximatif de transistors égal à 73958.

IX.2.3 : Entrées contrôlées

	nbr transistors	multiplicité	sous-tolal
comparateur de séquences	60	16	960
CDR	68	1	68
registres	14	54	756
LFSR	90	2	180
automate séquencement	330	2	660
compteur 9 bits	100	2	200
automate decomptage	300	2	600
TOTAL			3424

Tableau n°4 : Nombre de transistors des entrées

IX.2.4 : Sorties fréquence

	nbr transistors	multiplicité	sous-total
chaine de compteurs	3000	1	3000
interrupteur	6	9	54
registre	112	2	224
CDR	48	1	48
analyseur de signature	360	1	360
mémorisation d'erreur	80	1	80
TOTAL			3766

Tableau n° 5 : Nombre de transistors des sorties

Le total général du nombre de transistors du circuit s'élève donc à 101148 transistors. A partir de ce total on peut estimer qu'il faut rajouter 10% pour les transistors de liaison entre la partie contrôle et la partie opérative et les différents buffers de signaux et d'horloge. Le total est donc de **111262** transistors avec 76,5% de structures régulières (RAMs et ROMs).

Conformément au cahier des charges le boîtier retenu dispose de 84 sorties, et sachant que la taille d'un plot en technologie 1,5µm ES2 est de 150mm sur 584mm, nous obtenons une surface minimum de silicium exploitable de : 4,4 mm x 4,4 mm 19,36 mm^2 . Cette surface a été calculée en respectant une distance de 100µm entre deux soudures. Ce résultat nous permet de constater que si l'on veut conserver la surface minimum il est nécessaire de prévoir une intégration de 5800 transistors au mm^2. Ce chiffre peut vraisemblablement être atteint si l'on considère le taux de structures régulières et si l'on envisage de faire un layout avec une technologie de 1,2µm.

IX.3 CALCUL DE LA FIABILITE PREVISIONNELLE

Cette évaluation est effectuée à partir des données fournies par le CNET [Receuil de données de fiabilité du CNET] . Elle permet de calculer un MTBF du circuit et ainsi nous donner un ordre de grandeur de la disponibilité du système.

Les calculs ont été effectués à partir d'une estimation de la complexité du circuit. Ce calcul prend en compte la dualité des blocs et la hiérarchie des contrôles. C'est à dire que l'on prend en compte les contrôles effectués sur chaque bloc, puis les contrôles effectués entre un bloc et un bloc dual et enfin le contrôle global du circuit. La ROM externe est aussi prise en compte puisqu'elle influe directement sur la fiabilité du système.

Choix des paramètres

Pour faire les calculs, il est indispensable de choisir un certain nombre d'options en fonction des conditions d'exploitation du système.

Ces choix sont les suivants :

- Technologie	CMOS
- Boîtier	Etanche (environnement peu favorable)
- Tension maximale spécifiée	< 12 V
- Tension de service	<= 5V
- Température de jonction	70°C (circuit lent, contraintes faibles)
- Environnement	Matériel fixe au sol
- Qualifications	NFC D

Nous traiterons deux cas d'ancienneté. Une expérience de moins d'un an et une expérience de plus de deux ans.

Formule utilisée :

$$\lambda = [(C1.\pi T.\pi t.\pi v) + (C2.\pi B.\pi E.\pi s)]\pi L.\pi Q.10^{-9}\ h^{-1}$$

Avec :

- **C1** et **C2** liés à la complexité
- **πT** lié à la technologie du composant
- **πt** lié à la nature du boîtier
- **πv** lié à la tension d'utilisation, à la tension de spécification et à la température de la puce = 0.5
- **πB** lié à l'environnement et à la technologie = 1
- **πE** lié à l'environnement = 6
- **πs** lié au nombre de sorties
- **πL** lié à l'ancienneté du circuit
- **πQ** lié à la qualification du composant

Suivant les prescriptions du CNET, la partie opérative et la partie contrôle ont été décomposées en deux sous ensembles, l'un à motifs ordonnés, l'autre à structure anarchique.

La ROM externe ne fait pas partie intégrante du circuit MAPS, mais elle conditionne la fiabilité du système.

Partie contrôle , Micro-ROM + nano-ROM

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_s	π_L	π_Q
530	1,7	1,3	0,5	53	1	6	4	4 puis 1	0,8

λ h - 1	MTBF heures	MTBF années	NFC D
0,592 10-5	168918	19,28	<= 1 an
0,148 10-5	675675	77,1	>= 2 ans

Contrôleur double-rail associé à l'ensemble nano-ROM, micro-ROM

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_s	π_L	π_Q
46	20	1,4	0,5	13	1	6	1	4 puis 1	0,8

λ h - 1	MTBF heures	MTBF années	NFC D
0,23110-5	432825	49,40	<= 1 an
0,057 10-5	1731601	197,6	>= 2 ans

calcul de : λROM +λ CDR

λ R O M h - 1	MTBF heures	MTBF années	NFC D
0,823 10-5	121506	13,87	<= 1 an
0,20510-5	487804	55,68	>= 2 ans

Calcul de la fiabilité du reste de la partie contrôle (PLA + MUXs + automate)

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_s	π_L	π_Q
70	20	1,4	0,5	23	1	6	1	4 puis 1	0,8

λ h - 1	MTBF heures	MTBF années	NFC D
0,35710-5	280112	31,97	<= 1 an
0,0892 10-5	1120448	127,9	>= 2 ans

$\lambda PC = (\lambda ROM + \lambda\ CDR\ ) + \lambda(PLA + automate + MUX) = \lambda PC\ *$

Calcul de la fiabilité du circuit de contrôle des deux parties contrôles.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
63	20	1,4	0,5	16	1	6	1	4 puis 1	0,8

$\lambda\ h^{-1}$	MTBF heures	MTBF années	NFC D
0,31210-5	319529	36,47	<= 1 an
0,078 10-5	1282051	146,3	>= 2 ans

Calcul de $\lambda PCt = \lambda PC + \lambda PC^* + \lambda$contrôle

$\lambda PCth^{-1}$	MTBF heures	MTBF années	NFC D
2,672 10-5	37425	4,27	<= 1 an
0,668 10-5	149700	17,8	>= 2 ans

Calcul de la fiabilité de la RAM interne de 11500 transistors.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
390	3,5	1,3	0,5	49	1	6	1	4 puis 1	0,8

$\lambda\ RAM\ h^{-1}$	MTBF heures	MTBF années	NFC D
0,378 10-5	264550	30,19	<= 1 an
0,0945 10-5	1058201	120,8	>= 2 ans

Calcul du contrôle de la RAM.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
40	20	1,4	0,5	11	1	6	1	4 puis 1	0,8

λ h - 1	MTBF heures	MTBF années	NFC D
0,2 10-5	499201	56,98	<= 1 an
0,05 10-5	2000000	228,3	>= 2 ans

Calcul de la fiabilité de la RAM self-checking $\lambda RAMsc = \lambda RAM + \lambda contrôle$

$\lambda RAMsc\ h^{-1}$	MTBF heures	MTBF années	NFC D
0,578 10-5	173010	19,75	<= 1 an
0,144 10-5	692041	79	>= 2 ans

Calcul de la fiabilité de l'UAL

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
63	20	1,4	0,5	22	1	6	1	4 puis 1	0,8

$\lambda UAL\ h^{-1}$	MTBF heures	MTBF années	NFC D
0,324 10-5	308185	35,18	<= 1 an
0,081 10-5	1234567	140,9	>= 2 ans

Calcul de la fiabilité du générateur de test.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
33	20	1,4	0,5	17	1	6	1	4 puis 1	0,8

λ gené testh^{-1}	MTBF heures	MTBF années	NFC D
0,180 10-5	554078	62,25	<= 1 an
0,045 10-5	2222222	253,67	>= 2 ans

Calcul de la fiabilité du bloc des entrées.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
105	20	1,4	0,5	29	1	6	1	4 puis 1	0,8

λ entréesh^{-1}	MTBF heures	MTBF années	NFC D
0,526 10-5	190085	21,69	<= 1 an
0,1315 10-5	760456	86,81	>= 2 ans

Calcul du contrôle de la partie opérative et des entrées et de la ROM externe car tous ces blocs sont contrôlés par le contrôleur du bus.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
45	20	1,4	0,5	13	1	6	1	4 puis 1	0,8

λ contrôleh^{-1}	MTBF heures	MTBF années	NFC D
0,2265 10-5	441384	50,38	<= 1 an
0,0566 10-5	1766004	201,6	>= 2 ans

Calcul de la fiabilité de la ROM externe, c'est un ROM du commerce de 70000 transistors.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
830	1,7	1,3	0,5	63	1	6	2	4 puis 1	0,8

$\lambda ROMext h^{-1}$	MTBF heures	MTBF années	NFC D
0,5354 10-5	186776	21,31	<= 1 an
0,1338 10-5	747104	85,28	>= 2 ans

Calcul de la fiabilité de la RAM de communication double-accès

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
220	3,5	1,3	0,5	39	1	6	2	4 puis 1	0,8

$\lambda RAMCh^{-1}$	MTBF heures	MTBF années	NFC D
0,309 10-5	322663	36,83	<= 1 an
0,0772 10-5	1294498	147,7	>= 2 ans

Calcul de la fiabilité de la partie opérative complète. $\lambda OPt = \lambda OP + \lambda OP^* + \lambda RAMC + \lambda$contrôle et $\lambda OP = \lambda UAL + \lambda RAMsc + \lambda$géné + $\lambda ROMext + \lambda$entrées et $\lambda OP = \lambda OP^*$.

$\lambda OPth^{-1}$	MTBF heures	MTBF années	NFC D
4,82 10-5	20618	2,35	<= 1 an
1,2 10-5	83333	9,51	>= 2 ans

Calcul de la fiabilité des sorties.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
99	20	1,4	0,5	17	1	6	1	4 puis 1	0,8

λ sortiesh^{-1}	MTBF heures	MTBF années	NFC D
0,476 10-5	210013	23,97	<= 1 an
0,119 10-5	840336	94,9	>= 2 ans

Calcul de la fiabilité du circuit de contrôle

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
33	20	1,4	0,5	17	1	6	1	4 puis 1	0,8

λ ctrh^{-1}	MTBF heures	MTBF années	NFC D
0,180 10-5	554078	62,25	<= 1 an
0,045 10-5	2222222	253,67	>= 2 ans

λ sorties ctrh^{-1}	MTBF heures	MTBF années	NFC D
0,656 10-5	152439	17,4	<= 1 an
0,164 10-5	609756	69,6	>= 2 ans

La partie contrôle et la partie opérative (directe et duale) sont contrôlées par un CDR self exercising

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_S	π_L	π_Q
56	20	1,4	0,5	21	1	6	1	4 puis 1	0,8

λ CDRseh^{-1}	MTBF heures	MTBF années	NFC D
0,291 10-5	343406	39,2	<= 1 an
0,072 10-5	1374570	156,9	>= 2 ans

Calcul de la fiabilité de la mémorisation d'erreur.

C_1	π_T	π_t	π_v	C_2	π_B	π_E	π_s	π_L	π_Q
47	20	1,4	0,5	14	1	6	1	4 puis 1	0,8

λ mémo h^{-1}	MTBF heures	MTBF années	NFC D
0,237 10-5	421159	48	<= 1 an
0,0592 10-5	1687763	192,6	>= 2 ans

Calcul de la fiabilité du circuit MAPS: $\lambda MAPS = \lambda PCt + \lambda POt + \lambda sorties + \lambda mémo + \lambda CDRse$

λ MAPS h^{-1}	MTBF heures	MTBF années	NFC D
8,676 10-5	11523	1,31	<= 1 an
2,16 10-5	46296	5,28	>= 2 ans

Certains paramètres agissent directement sur les résultats que l'on vient de présenter, il s'agit notamment des paramètres liés à l'ancienneté des circuits et à la qualification de la technologie. A titre d'exemple, l'utilisation d'une technologie spatiale peut nous permettre de gagner un facteur 2,7 sur chacun des résultats exposés ci-dessus.

IX.4 PANNES CONTRAIRE A LA SECURITE

Une étude menée par S Noraz [NOR 89] sur une interface sensiblement identique à celle proposée pour le circuit MAPS fait apparaître le résultat suivant :
La probabilité d'apparition d'un évènement contraire à la sécurité à taux de défaillance constant et à intervalle de temps entre test de 1 minute est de $1,3\ 10^{-20}$.

On peut cependant remarquer que la masse reliée aux transistors T3 et T4 de l'interrupteur de la figure 9 du chapitre VIII est commune, ceci réduit le nombre d'évènements cumulés avant catastrophe à 3. L'utilisation de deux masses (ce qui peut facilement être réalisé dans le MAPS) ou l'ajout de transistors permettrait de porter le nombre d'évènements cumulés avant catastrophe à 4.

Dans le cas du MAPS, la phase de test hors-ligne est activée toutes les 100ms et le λ acquis théoriquement pour l'interface est de $0,65\ 10^{-5}\ h^{-1}$. En appliquant la simplification, $(1 - e^{-\lambda t}) = \lambda t$

ce qui est tout à fait réaliste devant l'ordre de grandeur des chiffres traités, et en considérant un ordre de 4, on obtient la probabilité d'apparition d'un évènement catastrophique égale à :

$$(0{,}65.10^{-5}\frac{0{,}1}{3600})^4 = 10^{-39}$$

Le circuit MAPS peut fournir des informations erronées non détectables s'il est soumis à l'apparition de 4 pannes avant la fin de la phase de test hors ligne. Si l'on considère le λ le plus défavorable (celui du circuit total) calculé théoriquement soit 8, 68 10^{-5} h^{-1} on obtient la probabilité suivante :

$$(8{,}68.10^{-5}\frac{0{,}1}{3600})^4 = 3.10^{-34}$$

Ce résultat est la limite théorique d'apparition d'un évènement dangereux généré par le MAPS. Ce chiffre n'a plus de signification réelle à l'échelle humaine, et même si l'écart entre la théorie et la réalité fait intervenir un facteur 10^{10} , le résultat obtenu reste tout à fait satisfaisant et de surcroit nettement supérieur aux résultat actuels.
Un tel niveau de sécurité du matériel sera en fait limité par le niveau de sécurité du logiciel de décodage des informations (code mathématique).
A la vue de ces chiffres on peut aussi prévoir que les pannes de mode commun (orages, surtension,...), capables de générer, en moins de 100 ms, une panne indétectable, (ordre 4), ne peuvent pas être considérées comme improbables. La limitation de la sécurité est donc atteinte par la probabilité d'apparition de ce type de panne. Une étude statistique (nombre d'orages et intensité, champ électrique, champ magnétique ...), sur l'emplacement même du circuit MAPS peut être envisagée mais les règles générales resteront difficiles à modéliser.

IX.5 TEMPS DU TEST

1 : Temps de communication

La phase d'émission/réception des trois messages, ainsi que la validation du message reçu et le codage du message émis, ont fait l'objet d'une description en langage machine (jeu d'instructions, plus quelques opérations internes). Cette description nous a permis d'estimer le temps de communication à 10,762 ms.

2 : Temps de test

Le temps de test de chacun des organes peut être estimé séparément :

- Contrôleur double-rail global	0,024ms
- Micro-ROM + nano-ROM	4,096ms
- PLA	0,032ms
- Unité arithmétique	0,512ms

- Unité logique	0,512ms
- CDR bus et sorties	0,032ms
- Test interrupteur	0,007ms
- Test commande UA	0,010ms
- Test RAM	2,176ms
- Test ROM externe	40,960ms
- Analyse de signature (estimation)	0,300ms

Le test de la sortie S9 peut être déclenché en même temps que celui de la micro-ROM.
Le test des entrées peut aussi être déclenché en parallèle, il suffit de bien choisir la dernière adresse générée lors du test de la micro-ROM. Cette adresse déclenche le test des entrées et leurs passages sur le bus (lecture). Cette adresse reste fixe pendant tout le temps du test du PLA soit 0,032ms alors que le test des entrées ne demande que 0,014ms. L'adresse suivante de cette adresse déclenchera le test d'un autre organe du circuit dès la fin du test du PLA.
En conséquence,on obtient un temps de test de l'ordre de 49,569ms.
L'instruction STOP nécessite donc 60,231ms, c'est à dire, qu'à chaque cycle,il ne reste que moins de 40ms pour l'interprétation. Cela est largement suffisant si l'on considère la simplicité des programmes à faire exécuter par le MAPS.

IX.6 CONCLUSION

Ce dernier chapitre nous a permis de donner une estimation des données principales du circuit avant de débuter une phase de conception. Ces chiffres sont, le nombre de transistors, la probabilité de défaillance des composants et les divers temps de test . Ces chiffres sont extraits des documents présentés en annexe, ils nous permettent de remarquer l'importance de la partie contrôle par rapport au reste du circuit. Cette disproportion est essentiellement due à l'instruction STOP qui représente 90% du remplissage de la partie contrôle.
La diminution de la taille du circuit passe donc par la simplification du codage des données ou par l'implémentation du programme de décodage en ROM externe. Cependant, cette seconde solution augmenterait le temps de traitement et entraînerait des modifications de l'architecture (taille du registre d'instructions).

CONCLUSION

Les nouvelles techniques de test intégré développées au sein du laboratoire TIM3 et, rappelées au début de cette étude, nous ont permis de présenter une étude de faisabilité originale en vue de réaliser un système complet de très haute sécurité pour l'application particulière qu'est la signalisation ferroviaire. Pour cela, nous avons conçu, pour la première fois, un circuit auto-contrôlable (self-checking) disposant de sorties, soit sûres soit correctes (fail-safe), dont l'architecture repose sur la technique UBIST associée à deux interfaces "strongly fail-safe" et ceci sur le même circuit.
On a ainsi associé un test en-ligne et un test hors-ligne très performants pour atteindre le TSC goal (détection de la première panne dès son apparition).
Le choix de la redondance hétérogène associée au code de parité (PO), ou à un code non ordonné, (PC) permet au circuit de détecter toutes les pannes triples ainsi qu'une très grande partie des pannes de multiplicité supérieure.
Le circuit assure donc que l'apparition d'une situation dangereuse est très inférieure à ce que pourraient atteindre les éléments externes au micro-contrôleur MAPS (transmission, relais ...). Cependant la probabilité d'apparition d'une panne de mode commun susceptible de modifier deux bits duaux et leurs parités respectives reste difficilement appréciable.

A l'heure actuelle, le développement des circuits auto-contrôlables passe par l'élaboration d'outils capables de générer automatiquement des blocs auto-contrôlables, car les bibliothèques font défaut, et le coût de conception s'en ressent. De plus, la validation des règles de conception, essentielle pour un circuit de sécurité, est longue et fastidieuse mais c'est une étape déterminante. La duplication hétérogène de blocs totalement indépendants semble permettre la validation la plus aisée. De plus, toutes les performances permettant d'améliorer la sécurité sont acquises en réduisant le coût et l'encombrement d'un système complet (dont l'unité de base est l'armoire) à ceux d'une simple carte.

La propriété d'autotestabilité peut trouver un champ d'application en dehors du domaine de sécurité pour ses potentialités concernant la maintenance. En particulier, l'automate peur être utilisé comme gestionnaire d'accès à des bus, en télécommunication et en téléinformatique. Les propriétés d'autotestabilité permettent d'assurer qu'un terminal défaillant se retire par lui même du service, ce qui éviterait la pollution du reste de la station. L'application la plus importante pourrait certainement être les réseaux numériques avec intégration de service RNIS, car tout terminal défaillant est susceptible de bloquer l'accès complet au bus, voire de perturber fortement le point d'accès à

l'intérieur du réseau public. Le problème est d'autant plus crucial pour des installations comportant des terminaux automatiques (télécopie) pour lesquels le diagnostique doit être effectué à distance.

Le microcontrôleur MAPS a aussi l'avantage d'utiliser un codage en fréquence au niveau de ses sorties, cette solution est très répandue pour commander des actionneurs tels que l'on pourrait les rencontrer dans une installation de centrale nucléaire ou tout autre industrie à hauts risques.
L'utilisation d'une micro-ROM de commande permet aussi d'envisager des modifications du micro-programme à moindre coût.

Le micro-contrôleur MAPS dispose d'une architecture du type micro-processeur mais ne dispose que d'un jeu d'instructions réduit et d'une fréquence de fonctionnement assez faible. Il semble donc raisonnable de penser qu'un développement de ce circuit le ferait évoluer vers un microprocesseur d'application plus générale et totalement auto-contrôlable. La meilleure solution restant la conception d'un petit micro-processeur original, et non pas la modification d'un micro-processeur existant .

Cette étude a donc avant tout permis de monter qu'il existait des solutions au problème de sécurité des circuits complexes pour des applications critiques, mais aussi de constater le chemin qu'il restait à parcourir avant d'exploiter de tels systèmes. Une version moins performante a déjà fait l'objet de plusieurs présentations [CHA 90a], [CHA 90b],[CHA 90c].

Toutefois, le champ d'application et les développements potentiels d'un tel circuit permettent de penser que la technique employée devrait se généraliser dans un avenir proche. Sachant très bien que la perfection ne sera jamais atteinte, il faudra donc toujours évoluer entre maintenabilité, fiabilité et sécurité, en étant conscient que le point le plus faible donnera toujours la limite supérieure de la qualité du sytème considéré. Ce qui revient à dire qu'il est inutile d'avoir ponctuellement des qualités très supérieures à la moyenne, par exemple au niveau du circuit, si le système général (ligne de transmission, logiciel ...) a des qualités très ordinaires.

REFERENCES

ABO 84 Aboulhamid M, Cerny E "Built-in testing of one-dimensional unilateral iterative arrays" IEEE trans. computer Vol C-33 June 84

AND 71 Anderson D.A "design of self checking digital networks using coding technique" - coorditated science laboratory report p 527 univ of illinois urbana sep 71.

CAR 68 Carter W.C, Scheider P.R "Design of dynamically checked computers" IFIP Congress, Edinburg 1968 Inf.Proc. 68 Amsterdam, North Holland 1969

CAS 89 Castro Alves V "Participation à l'étude d'un microprocesseur de très haute sécurité de fonctionnement destiné aux transport ferroviaire" rapport de DEA juin 89

CER 88 Cerny E, Aboulhamid M, Bois G, Cloutier G "Built-in self-test of a CMOS ALU" IEEE Design& test of computer August 88

CHA 90a Chaumontet G, Nicolaidis M, Guyot A, Castro Alves V, Courtois B "Description d'un micro-automate programmable de sécurité (MAPS) à l'usage de la signalisation ferroviaire" 7 ième colloque international de fiabilité et de maintenabilité Brest juin 90

CHA 90b Chaumontet G,Castro Alves V, Nicolaidis M, Guyot A, Courtois B " MAPS : a safety microcontroller dedicated to railway control" FTSD - 13 Varna Bulgaria june 90

CHA 90c Chaumontet G,Castro Alves V, Nicolaidis M, Guyot A, Courtois B " A fail-safe microcontroller for railway signalling" ESSCIRC'90 Grenoble France september 90

CHI 82 Chiang K.W "test generation for MOS complex gate networks" 12 th fault tolerant computing symposium Santa Monica June 82 USA

COR 89 Cornil J.P "Conception de la partie opérative d'un micro-automate de sécurité" Mémoire d'ingénieur Juin 89

COU 81 Courtois B "failure mechanisms, fault hypothese and analytical testing of LSI_NMOS (HMOS) circuits.VLSI 81 univ of Edinbourg au 18/21 81.

DAR 89 Darley F "Contribution au test des circuits intégrés CMOS: Etude du test des pannes stuck-on et stuck-open" Thèse de Doct INPG Nov 89

DAV 78 David R, Thevenod-Fosse P "Design of totally self-checking asynchronous modular circuits" Journal of Design Automation and Fault Tolerant Computing Vol 2 oct 78

ELZ 81 Elziq Y M automatic test generation for stuck-open fault in CMOS LSI, 18th design automatic conf june 81 Nashville.

FER 88 Ferguson F.J, Shen J.P "Extraction and simulation of realistic C-MOS faults using inductive fault analysis" ITC 88

FRI 71 Friedman A.D "Fault detection in digital circuits" Prentice Hall Inc 71

FRI 73 Friedman A.D "Easily testable iterative systems" IEEE trans comput vol C-22 Dec 73

FUJ 85 Fujiwara H "Logic testing and design for testability" MIT Press series in computer systems Cambridge 85

FUT 88 Futsuhara K, Sugimoto N, Mukaidono M "Fail-safe logic element having upper and lower thresholds and their application to safety control". The 18th Int Symp on Fault-Tolerant Computing Tokyo, june 1988.

GAL 80 Galiay J Al, Crouzet Y, Vergniault M . physical versus logical fault models MOS LSI circuits; impact on their testability IEEE trans on comp , vol c-29 n°6 june 80

JHA 88 Jha N. K "SFS/SSC domino CMOS implementation of TSC circuits" in proc Annual Allerton Conf Communication Control&Comput Los Angeles Allerton IL pp768- 777 sept 88

JHA 89 Jha N.K " Fault detection in CVS parity trees application to SCC/CVS parity and two rail checkers", in Proc Int Symp Fault-tolerant Comput Chicago IL, pp407 - 414 june 90

KLO 87 Kloppenburg T " Logisire, a safe computer system for process-automation" Fehlentolerende rechensysteme 3 rd International GI/ITG/GMA conference, Bremerhaven 9-11 sep 1987

MAK 82 Mak G.P, Abraham J.A, Davidson E.S "The design of PLAs with concurrent error detection" 12th Int symp On Fault Tolerant Comp Santa Monica June 82

MAL 82 Malaiya Y K , Su S Y H a new fault model and testing technics for CMOS devices

MAR 82 Marinescu M "simple and efficient algorithmes for functional RAM testing" IEEE International Test Conference novembre 82

MIN 67 Mine H, Koga Y "Basic properties and a construction method for fail-safe logical systems" IEEE Trans Elec Comp june 1967.

NAN 87 Nanya T, Kawamura T "A note on strongly fault secure sequential circuits" IEEE Trans Comp Vol C-36 Sept 87

NAN 88 Nanya T, Kawamura T "Error secure/propagating concept and its application to the design of strongly fault secure processors" IEEE Trans Comp Vol C-37 January 88

NAN 89 Nanya T, Uchida M " A strongly fault secure and strongly code disjoint realization of combinational circuits" 19th Int Symp On f-Fault Tolerant Comp Chicago june 89

NIC 84a Nicolaidis M, "conception de circuits intègrès autotestables pour des hypothéses de pannes analytiques" thése de docteur-ingenieur jan 84

NIC 84b Nicolaidis M , Jansch J.C, Courtois B "Stongly code disjoint checker" the 14th Int symp on fault tolerant comp Kissimmee june 1984 new york ,IEEE 1984 p 16-21 and IEEE trans on comp june 1988.

NIC 85a Nicolaidis M, Courtois B "layout rules for the design of self-checking circuits" VLSI conf august 85 tokyo

NIC 85b Nicolaidis M "evaluation of a self-checking version of the MC 68000 microprocessor proceeding " 15th fault tolerant computing symposium Ann Arbor june 85 USA

NIC 86 Nicolaidis M , Courtois B "design of self-checking circuits using unidirectional error detecting codes" 16th fault tolerant computing symposium Vienne july 86

NIC 87 Nicolaidis M "shorts in self-checking circuits" proc IEEE 87 int'l test conference

NIC 88 Nicolaidis M "a unified built-in self test scheme UBIST" 18th fault tolerant computing symposium tokyo june 88 and IEEE Transaction on CAD/ICAD

NIC 89 Nicolaidis M, Noraz S, Courtois B. "A generalized theory of fail-safe systems" The 19th Int Symp.On Fault Tolerant Comp (FTCS)CHICAGO june 89

NIC 90a Nicolaidis M "Test pattern generators for arithmetic units and arithmetic and logic units" rapport de recherche janvier 90

NIC 90b Nicolaidis M "Efficient UBIST implementation microprocessor sequencing parts" International test conference ITC 1990 Washington

NOR 89 Noraz S "Application des circuits intégrés autotestables à la sécurité de fonctionnement des systèmes" Thèse de l'INPG décembre 89

OKL 84 Oklobddzija V . Kovijanic p "On testability of CMOS-Domino Logic" Proc Fault Tolerant Computing Symp June 84 pp 50 - 55

SAL 75 Salomon G, Therond J.C "Systèmes de logique dynamique pour les circuits de protection des réacteurs nucléaires" Journées d'information des industries nucléaires "NUCLEX 75"

SMI 78 Smith J E , Metze G "strongly fault-secure logic networks" IEEE trans comp vol C-27 n°6 june 78

SRI 81 Sridhar T, Hayes J.P "Design of easily testable bit sliced systems" IEEE Trans Computers Vol C-30 Nov 81

TAK 71 Takaoka T, Mine H "N-fail-safe logical systems" IEEE Trans Comp Vol C-20 may 71

THO 87 Thorel P, David R, Pulou J, Raiward J.L "Design for random testability" ITC Washington D.C sept 87 USA

WAD 78 Wadsack R L "fault modelling and logic simulation of CMOS and MOS integreted circuits" the bell system tecnical journal may 78.

WES 85 Weste N.H.E and Eshraghian K "principles of CMOS VLSI Design : A system perspective" Addison Wesley Publ Co 85

ANNEXES

ANNEXE I

Description en langage machine des opérations permettant d'effectuer la procedure d'émission et de réception.
Cette description a été traduite d'un programme en langage C.

L'intérêt de cette description a été avant tout de pouvoir déterminer la taille de la partie contrôle et d'estimer le temps de traitement. La validation de cette procédure n'a pas été faite par l'industriel et il manque un certain nombres d'informations. Cependant, il n'est de toute façon pas utile pour l'étude de faisabilité de disposer d'exactement toutes les données. Cette partie est de loin la plus importante à cause de la compléxité du codage.
De la même manière, on trouvera un description des opérations permettant d'effectuer la procédure de test. On pourra remarquer que l'algorithme de Marinescu demande plus de la moitié de cette description car dans la plupart des cas, le test consiste à déclencher un générateur et à attendre le compte rendu de fin de test.
A titre de comparaison, on trouvera ensuite la description du fonctionnement normal qui ne représente qu'à peu près 10% du remplissage total du plan mémoire.

adresse courante	adresse suivante	commande	phase	test	commentaires
n	n+1	ram(2) <- A	3	0	sauvegarde
n+1	n+2	ram(3) <- C	3	0	
n+2	n+3	A <- COl	3	0	
n+3	n+4	ram(50) <- A	3	0	
n+4	n+5	A <- COh	3	0	
n+5	n+6	ram(51) <- A	3	0	
n+6	n+7	A <- 0	3	0	initialisations des
n+7	n+8	A(0) <- C=1	3	0	compteurs du
n+8	n+9	M(25) <- A	3	0	nombre de passages
n+9	n+10	M(4) <- A=0	3	0	
n+10	n+11	M(21) <- A=0	3	0	
n+11	n+12	A <- 0	3	0	
n+12	n+13	A(3) <- C=1	3	0	
n+13	n+14	A(5) <- C=1	3	0	
n+14	n+15	M(22) <- A	3	0	M(22) = 40
n+15	n+16	A <- 0	3	0	
n+16	n+17	A(3) <- C=1	3	0	
n+17	n+18	A(1) <- C=1	3	0	M(23) = 10
n+18	n+19	M(23) <- A	3	0	
n + 19	n + 20	A(2) <- C = 1	3	0	
n + 20	n + 21	A(3) <- C = 1	3	0	
n + 21	n + 22	A(4) <- C = 1	3	0	
n + 22	n + 23	A(5) <- C = 1	3	0	A = 60
n + 23	n + 24	M(55) <- A	3	0	
n + 24	n + 25	RegRamc <- M(55) A<-Ramc(M(55))	2-3	0	
n + 25	n + 27	C <- A(0)	3	1	
C = 0 -> n + 26	n + 39	NOP		1	
C = 1 -> n + 27	n + 28	B <- M(22) et RI <- B + 0 + 0	2-3	0	RECEPTION
n + 28	n+29	RegRamc <- M(21)	2	0	(intr = 1)
n + 29	n+30	M(RI) <- Ramc(M(21))	3	0	
n+30	n+31	M(22) <- M(22) + 1	2-3	0	
n+31	n+32	M(21) <- M(21) + 1	2-3	0	
n+32	n+34	A <- M(23) + FF	2-3	1	
C = 0 -> n + 33	n+35	A <- M(4)	3	0	
C = 1 -> n + 34	n+27	NOP		0	
n+35	n+36	A <- A + 1	3	0	
n+36	n+38	C <- A(1)	3	1	
C = 0 -> n + 37	n+41	A <- M(60)	3	0	
C = 1 -> n + 38	n+106	A <- 0	3	0	
comp=0 -> n+39	n+18	NOP		0	
comp=1 -> n+40	n+411	ERREUR			
n+41	n+42	M(56) <- A	3	0	duplication date
n+42	n+43	A <- M(61)	3	0	
n+43	n+44	M(57) <- A	3	0	
n+44	n+45	A <- M(62)	3	0	
n+45	n+46	M(58) <- A	3	0	
n+46	n+47	A <- M(63)	3	0	
n+47	n+48	M(59) <- A	3	0	
n+48	n+49	A <- M(40)	3	0	
n+49	n+51	A <- A(x)	3	1	
C=0 -> n+50	n+52	A <- M(41)	3	0	
C=1 -> n+51	n+59	A <- M(41)	3	0	
n+52	n+53	M(60) <- A	3	0	traitement
n+53	n+54	A <- M(42)	3	0	message date

n+54	n+55	M(61) <- A	3	0	
n+55	n+56	A <- M(43)	3	0	
n+56	n+57	M(62) <- A	3	0	
n+57	n+58	A <- M(44)	3	0	
n+58	fin n+414	M(63) <- A	3	0	
n+59	n+60	M(31) <- A	3	0	traitement
n+60	n+61	A(7) <- C=0	3	0	message info
n+61	n+62	A(6) <- C=0	3	0	élimination bits PF
n+62	n+63	A(5) <- C=0	3	0	du 1er octet
n+63	n+64	A(4) <- C=0	3	0	
n+64	n+65	M(24) <- A	3	0	
n+65	n+66	A <- M(56)	3	0	
n+66	n+67	A(7) <- C=0	3	0	élimination bits PF
n+67	n+68	A(6) <- C=0	3	0	de l'octet pf de date
n+68	n+69	A(5) <- C=0	3	0	
n+69	n+70	A(4) <- C=0	3	0	
n+70	n+71	B <- M(24), A <- M(24)-A-C	2-3	0	calcul écart date
n+71	n+73	C <- A(4)	3	1	
C=0 -> n+72	n+85	C <- A(3)	3	0	
C=1 -> n+73	n+74	C <- A(3)	3	0	
n+74	n+76	C <- C ET C'	3	1	
C=0 -> n+75	n+77	C <- A(2)	3	0	
C=1 -> n+76	n+412	ERREUR			écart date >>8
n+77	n+79	C <- C OU C'	3	1	
C=0 -> n+78	n+105	NOP	3	0	date inchangée
C=1 -> n+79	n+80	A <- 0	3	0	
n+80	n+81	A(3) <- C=1	3	0	
n+81	n+82	M(56) <- M(56) + A + C	2-3	0	date = date + 8
n+82	n+83	M(57) <- M(57) + 0 + C	2-3	0	
n+83	n+84	M(58) <- M(58) + 0 + C	2-3	0	
n+84	n+97	M(59) <- M(59) + 0 + C	2-3	0	
n+85	n+87	C <- C OU C'	3	1	
C=0 -> n+86	n+412	ERREUR			écart date << -8
C=1 -> n+87	n+88	C <- A(2)	3	0	
n+88	n+90	C <- C ET C'	3	1	
C=0 -> n+89	n+91	A <- 0	3	0	
C=1 -> n+90	n+105	NOP	3	0	date inchangée
n+91	n+92	A(3) <- C=1	3	0	
n+92	n+93	M(56) <- M(56) - A - C	2-3	0	date = date - 8
n+93	n+94	A <- 0	3	0	
n+94	n+95	M(57) <- M(57) - A - C	2-3	0	
n+95	n+96	M(58) <- M(58) - A - C	2-3	0	
n+96	n+97	M(59) <- M(59) - A - C	2-3	0	
n+97	n+98	A <- M(56)	3	0	
n+98	n+99	M(60) <- A	3	0	
n+99	n+100	A <- M(57)	3	0	
n+100	n+101	M(61) <- A	3	0	
n+101	n+102	A <- M(58)	3	0	
n+102	n+103	M(62) <- A	3	0	
n+103	n+104	A <- M(59)	3	0	
n+104	n + 105	M(63) <- A	3	0	

n+105	n+106	A <- 0	3	0	acquisition de Ax1
n+106	n+107	A(3) <- C=1	3	0	et Ax2
n+107	n+108	COh <- A	3	0	
n+108	n+109	COl <- 0	3	0	
n+109	n+110	M(24) <- A	3	0	M(24) = 8
n+110	n+111	A <- 0	3	0	
n+111	n+112	A(5) <- C=1	3	0	
n+112	n+113	M(23) <- A	3	0	M(23) = 32
n+113	n+114	RI <- A	3	0	
n+114	n+115	M(32) <- ROM	2-3	0	
n+115	n+116	COl <- COl + 1	2-3	0	
n+116	n+117	COh <- COh + C"	2-3	0	
n+117	n+118	A <- 0	3	0	
n+118	n+120	M(24) <- M(24) - A - C	2-3	1	
C=0 -> n+119	n+121	A <- 0	3	0	
C=1 -> n+120	n+113	A <- M(23) + 0 + 1	2-3	0	
n+121	n+122	A(5) <- C=1	3	0	
n+122	n+123	A(4) <- C=1	3	0	
n+123	n+124	A(3) <- C=1	3	0	
n+124	n + 125	M(24) <- A	3	0	M(24) = 56
n+125	n+126	A <- M(32)	2-3	0	date = date. 2E56
n+126	n+127	A <- M(56) - A - C	2-3	0	modulo AX1
n+127	n+128	M(13) <- A	3	0	
n+128	n+129	A <- M(33)	2-3	0	
n+129	n+130	A <- M(57) - A - C	2-3	0	
n+130	n+131	M(14) <- A	3	0	
n+131	n+132	A <- M(34)	2-3	0	
n+132	n+133	A <- M(58) - A - C	2-3	0	
n+133	n+134	M(15) <- A	3	0	
n+134	n+135	A <- M(35)	2-3	0	
n+135	n+136	A <- M(59) - A - C	2-3	0	
n+136	n+138	M(16) <- A	3	1	
C=0 -> n+137	n+146	NOP			
C=1 -> n+138	n+139	A <- M(13)	2-3	0	
n+139	n+140	M(56) <- A	3	0	
n+140	n+141	A <- M(14)	2-3	0	
n+141	n+142	M(57) <- A	3	0	
n+142	n+143	A <- M(15)	2-3	0	
n+143	n+144	M(58) <- A	3	0	
n+144	n+145	A <- M(16)	2-3	0	
n+145	n+146	M(59) <- A	3	0	
n+146	n+147	A <- M(56)	2-3	0	
n+147	n+148	A <- M(56) + A + C	2-3	0	
n+148	n+149	A <- M(57)	2-3	0	
n+149	n+150	A <- M(57) + A + C	2-3	0	
n+150	n+151	A <- M(58)	2-3	0	
n+151	n+152	A <- M(58) + A + C	2-3	0	
n+152	n+153	A <- M(59)	2-3	0	
n+153	n+154	A <- M(59) + A + C	2-3	0	
n+154	n+155	A <- 0	3	0	
n+155	n+157	M(24) <- M(24) - A - C	3	1	
C=0 -> n+156	n+158	A <- M(60)	3	0	
C=1 -> n+157	n+126	A <- M(32)	3	0	
n+158	n+159	M(56) <- A	3	0	
n+159	n+160	A <- M(61)	2-3	0	
n+160	n+161	M(57) <- A	3	0	

n+161	n+162	A <- M(62)	2-3	0	
n+162	n+163	M(58) <- A	3	0	
n+163	n+164	A <- M(63)	2-3	0	
n+164	n+165	M(59) <- A	3	0	
n+165	n+166	A <- 0	3	0	
n+166	n+167	A(5) <- C=1	3	0	
n+167	n+168	A(4) <- C=1	3	0	
n+168	n+169	A(3) <- C=1	3	0	
n+169	n+170	M(24) <- A	3	0	M(24) = 56
n+170	n+171	A <- M(36)	2-3	0	date = date.2E56
n+171	n+172	A <- M(56) - A - C	2-3	0	modulo Ax2
n+172	n+173	M(17) <- A	3	0	
n+173	n+174	A <- M(37)	2-3	0	
n+174	n+175	A <- M(57) - A - C	2-3	0	
n+175	n+176	M(18) <- A	3	0	
n+176	n+177	A <- M(38)	2-3	0	
n+177	n+178	A <- M(58) - A - C	2-3	0	
n+178	n+179	M(19) <- A	3	0	
n+179	n+180	A <- M(39)	2-3	0	
n+180	n+181	A <- M(59) - A - C	2-3	0	
n+181	n+183	M(20) <- A	3	1	
C=0 -> n+182	n+191	NOP			
C=1 -> n+183	n+184	A <- M(17)	2-3	0	
n+184	n+185	M(56) <- A	3	0	
n+185	n+186	A <- M(18)	2-3	0	
n+186	n+187	M(57) <- A	3	0	
n+187	n+188	A <- M(19)	2-3	0	
n+188	n+189	M(58) <- A	3	0	
n+189	n+190	A <- M(20)	2-3	0	
n+190	n+191	M(59) <- A	3	0	
n+191	n+192	A <- M(56)	2-3	0	
n+192	n+193	A <- M(56) + A + C	2-3	0	
n+193	n+194	A <- M(57)	2-3	0	
n+194	n+195	A <- M(57) + A + C	2-3	0	
n+195	n+196	A <- M(58)	2-3	0	
n+196	n+197	A <- M(58) + A + C	2-3	0	
n+197	n+198	A <- M(59)	2-3	0	
n+198	n+199	A <- M(59) + A + C	2-3	0	
n+199	n+200	A <- 0	3	0	
n+200	n+202	M(24) <- M(24) - A - C	3	1	
C=0 -> n+201	n+203	A <- 0	3	0	
C=1 -> n+202	n+170	NOP			
n+203	n+204	A <- 255	3	0	acquisition de la
n+204	n+205	A(0) <- C=0	3	0	somme des contrôl
n+205	n+206	M(52) <- A	3	0	M(52) = 254
n+206	n+207	COh <- 0	3	0	
n+207	n+208	A <- M(31)	2-3	0	acquisition de CX1
n+208	n+209	COl <- A	3	0	dans le 1er Koctet
n+209	n+210	COl <- COl + A + (C"=0)	2-3	0	
n+210	n+211	COh <- COh + C"	2-3	0	
n+211	n+212	A <- COl	2-3	0	
n+212	n+213	COl <- COl + A + (C"=0)	2-3	0	
n+213	n+214	A <- COh	2-3	0	
n+214	n+215	COh <- COh + A + C"	2-3	0	
n+215	n+216	M(5) <- ROM	2-3	0	
n+216	n+217	COl <- COl + 1	2-3	0	
n+217	n+218	COh <- COh + C"	2-3	0	

n+218	n+219	M(6) <- ROM	2-3	0	
n+219	n+220	COl <- COl + 1	2-3	0	
n+220	n+221	COh <- COh + C"	2-3	0	
n+221	n+222	M(7) <- ROM	2-3	0	
n+222	n+223	COl <- COl + 1	2-3	0	
n+223	n+224	COh <- COh + C"	2-3	0	
n+224	n+225	M(8) <- ROM	2-3	0	
n+225	n+226	A <- COh	2-3	0	acquisition de CX2
n+226	n+227	A(2) <- C=1	3	0	dans le 2ièm Kocte
n+227	n+228	M12) <- ROM	2-3	0	
n+228	n+229	A <- 0	3	0	
n+229	n+230	COl <- COl - A - C"	2-3	0	COl = COl + FF
n+230	n+231	COh <- COh - A - C"	2-3	0	
n+231	n+232	M(11) <- ROM	2-3	0	
n+232	n+233	COl <- COl - A - C"	2-3	0	
n+233	n+234	COh <- COh - A - C"	2-3	0	
n+234	n+235	M(10) <- ROM	2-3	0	
n+235	n+236	COl <- COl - A - C"	2-3	0	
n+236	n+237	COh <- COh - A - C"	2-3	0	
n+237	n+238	M(9) <- ROM	2-3	0	
n+238	n+239	A <- M(13)	2-3	0	S1 = CX1 - codedat
n+239	n+240	M(5) <- M(5) - A - C	2-3	0	
n+240	n+241	A <- M(14)	2-3	0	
n+241	n+242	M(6) <- M(6) - A - C	2-3	0	
n+242	n+243	A <- M(15)	2-3	0	
n+243	n+244	M(7) <- M(7) - A - C	2-3	0	
n+244	n+245	A <- M(16)	2-3	0	
n+245	n+246	M(8) <- M(8) - A - C	2-3	0	
n+246	n+247	A <- M(17)	2-3	0	S2 = CX2 - codedat
n+247	n+248	M(9) <- M(9) - A - C	2-3	0	
n+248	n+249	A <- M(18)	2-3	0	
n+249	n+250	M(10) <- M(10) - A - C	2-3	0	
n+250	n+251	A <- M(19)	2-3	0	
n+251	n+252	M(11) <- M(11) - A - C	2-3	0	
n+252	n+253	A <- M(20)	2-3	0	
n+253	n+254	M(12) <- M(12) - A - C	2-3	0	
n+254	n+256	M(52) <- M(52) + 1	2-3	1	
C=0 -> n+255	n+257	A <- M(9)	2-3	0	
C=1 -> n+256	n+326	NOP			
n+257	n+258	M(5) <- M(5) + A + C	2-3	0	S = S1 + S2
n+258	n+259	A <- M(10)	2-3	0	
n+259	n+260	M(6) <- M(6) + A + C	2-3	0	
n+260	n+261	A <- M(11)	2-3	0	
n+261	n+262	M(7) <- M(7) + A + C	2-3	0	
n+262	n+263	A <- M(12)	2-3	0	
n+263	n+264	M(8) <- M(8) + A + C	2-3	0	
n+264	n+265	A <- M(46)			calcul de S' conten
n+265	n+266	M(42) <- M(42) + A + C	2-3	0	ue dans message
n+266	n+267	A <- M(47)	2-3	0	reçu
n+267	n+268	M(43) <- M(43) + A + C	2-3	0	
n+268	n+269	A <- M(48)	2-3	0	
n+269	n+270	M(44) <- M(44) + A + C	2-3	0	

n+270	n+271	A <- M(49)	2-3	0	
n+271	n+272	M(45) <- M(45) + A + C	2-3	0	
n+272	n+273	A <- M(42)	2-3	0	calcul de S - S'
n+273	n+274	M(5) <- M(5) - A - C	2-3	0	
n+274	n+275	A <- M(43)	2-3	0	
n+275	n+276	M(6) <- M(6) - A - C	2-3	0	
n+276	n+277	A <- M(44)	2-3	0	
n+277	n+278	M(7) <- M(7) - A - C	2-3	0	
n+278	n+279	A <- M(45)	2-3	0	
n+279	n+280	M(8) <- M(8) - A - C	2-3	0	
n+280	n+281	A <- 0	3	0	adressage de KC
n+281	n+282	A(3) <- C=1	3	0	
n+282	n+283	COh <- A	3	0	3ième Koctet
n+283	n+284	A <- 0	3	0	
n+284	n+285	A(3) <- C=1	3	0	
n+285	n+286	COl <- A	3	0	9ième octet
n+286	n+287	M(9) <- ROM	2-3	0	KC faible
n+287	n+288	COl <- COl + 1	2-3	0	
n+288	n+289	COh <- COh + C"	2-3	0	
n+289	n+290	M(10) <- ROM	2-3	0	KC moyen faible
n+290	n+291	COl <- COl + 1	2-3	0	
n+291	n+292	COh <- COh + C"	2-3	0	
n+292	n+293	M(11) <- ROM	2-3	0	KC moyen fort
n+293	n+294	COl <- COl + 1	2-3	0	
n+294	n+295	COh <- COh + C"	2-3	0	
n+295	n+296	M(12) <- ROM	2-3	0	KC fort
n+296	n+297	A <- M(9)	2-3	0	calcul de S - KC
n+297	n+298	M(5) <- M(5) - A - C	2-3	0	
n+298	n+299	A <- M(10)	2-3	0	
n+299	n+300	M(6) <- M(6) - A - C	2-3	0	
n+300	n+301	A <- M(11)	2-3	0	
n+301	n+302	M(7) <- M(7) - A - C	2-3	0	
n+302	n+303	A <- M(12)	2-3	0	
n+303	n+305	M(8) <- M(8) - A - C	2-3	1	
C=0 -> n+304	n+413	ERREUR			
C=1 -> n+305	n+306	A <- M(5)	2-3	0	
n+306	n+307	B <- M(6), A <- M(6) + A + C	2-3	0	
n+307	n+308	B <- M(7), A <- M(7) + A + C	2-3	0	
n+308	n+309	B <- M(8), A <- M(8) + A + C	2-3	0	
n+309	n+310	M(53) <- A	3	0	
n+310	n+311	A <- 0	3	0	
n+311	n+312	A <- 255	3	0	
n+312	n+314	B<-M(53), A<-M(53) + A + C	2-3	1	
C=0 -> n+313	n+315	M(29) <- RE	3	0	RE registre entrée1
C=1 -> n+314	n+413	ERREUR			
n+315	n+316	A <- M(60)			EMISSION
n+316	n+317	A(7) <- C=0	3	0	éliminer date PF de
n+317	n+318	A(6) <- C=0	3	0	l'octet pf
n+318	n+319	A(5) <- C=0	3	0	
n+319	n+320	A(4) <- C=0	3	0	
n+320	n+321	A(x1) <- C=1	3	0	
n+321	n+322	A(x2) <- C=1	3	0	indication INFO
n+322	n+323	M(40) <- A			
n+323	n+324	COh <- 0			

n+324	n+325	A <- M(29)			
n+325	n+208	M(41) <- A			acquerrir CXi
n+326	n+327	A <- M(5)	2-3	0	
n+327	n+328	M(42) <- A	3	0	
n+328	n+329	A <- M(6)	2-3	0	
n+329	n+330	M(43) <- A	3	0	
n+330	n+331	A <- M(7)	2-3	0	
n+331	n+332	M(44) <- A	3	0	
n+332	n+333	A <- M(8)	2-3	0	
n+333	n+334	M(45) <- A	3	0	
n+334	n+335	A <- M(9)	2-3	0	
n+335	n+336	M(46) <- A	3	0	
n+336	n+337	A <- M(10)	2-3	0	
n+337	n+338	M(47) <- A	3	0	
n+338	n+339	A <- M(11)	2-3	0	
n+339	n+340	M(48) <- A	3	0	
n+340	n+341	A <- M(12)	2-3	0	
n+341	n+343	M(49) <- A	3	0	
n+343	n+344	A <- M(25)	2-3	0	
n+344	n+346	C <- A(0)	3	1	
C=0 -> n+345	n+402	NOP			
C=1 -> n+346	n+347	C <- A(1)	3	0	
n+347	n+349	C <- C ET C'	3	1	
C=0 -> n+348	n+350	A <- 0	3	0	
C=1 -> n+349	n+406	NOP			
n+350	n+351	A(4) <- C=1	3	0	
n+351	n+352	A(3) <- C=1	3	0	
n+352	n+353	A(2) <- C=1	3	0	
n+353	n+354	A(1) <- C=1	3	0	
n+354	n+355	M(21) <- A	3	0	M(21) = 30
n+355	n+356	A <- 0	3	0	
n+356	n+357	A(5) <- C=1	3	0	
n+357	n+358	A(3) <- C=1	3	0	
n+358	n+359	M(22) <- A	3	0	M(22) = 40
n+359	n+360	A <- 0	3	0	
n+360	n+361	A(3) <- C=1	3	0	
n+361	n+362	A(1) <- C=1	3	0	
n+362	n+363	M(23) <- A	3	0	
n + 364	n + 365	RegRamc <- M(55) A<-Ramc(M(55))	2-3	0	
n + 365	n + 367	C <- A(0)	3	1	
C=0 -> n+366	n+384	NOP		1	EMISSION
C=1 -> n+367	n+368	B <- M(22) et RI <- B + 0 + 0	2-3	0	
n+368	n+369	RegRamc <- M(21)	2		
n+369	n+370	Ramc(M(21)) <- RAM(RI)	2	0	
n+370	n+371	M(22) <- M(22) + 1	2-3	0	
n+371	n+372	M(21) <- M(21) + 1	2-3	0	
n+372	n+373	A <- 0	3	0	
n+373	n+375	A <- M(23) + FF	2-3	1	
C=0 -> n+374	n+376	M(25) <- M(25) + 1	2-3	0	
C=1 -> n+375	n+364	NOP		0	
n+376	n+378	C <- A(2)	3	1	
C=0 -> n+377	n+379	C <- A(0)	3	0	
C=1 -> n+378	fin n+414	NOP	3	0	
n+379	n+380	A <- 0	3	0	
n+380	n+381	A <- 255	3	0	

n+381	n+383	M(4) <- A	3	1	
C=0 -> n+382	n+386	A <- M(31)	3	0	
C=1 -> n+383	n+394	A <- M(31)	3	0	
comp=0 ->n+384	n+362	NOP		0	
comp=1 ->n+385	n+411	ERREUR			
n+386	n+387	M(30) <- A	3	0	reception du 2ième
n+387	n+388	A <- M(29)	2-3	0	message
n+388	n+389	M(28) <- A	3	0	dans 10-19
n+389	n+390	M(29) <- RE2	3	0	
n+390	n+391	A <- 0	3	0	
n+391	n+392	A(3) <- C=1	3	0	
n+392	n+393	A(1) <- C=1	3	0	
n+393	n+11	M(21) <- A	3	0	
n+394	n+395	M(27) <- A	3	0	reception du 3ième
n+395	n+396	A <- M(29)	2-3	0	message
n+396	n+397	M(26) <- A	3	0	dans 20-29
n+397	n+398	M(29) <- RE3	3	0	
n+398	n+399	A <- 0	3	0	
n+399	n+400	A(4) <- C=1	3	0	
n+400	n+401	A(2) <- C=1	3	0	
n+401	n+11	M(21) <- A	3	0	
n+402	n+403	A <- 0	3	0	émission 2ième
n+403	n+404	A(5) <- C=1	3	0	message
n+404	n+405	A(3) <- C=1	3	0	dans 40-49
n+405	n+355	M(21) <- A	3	0	
n+406	n+407	A <- 0	3	0	émission 3ième
n+407	n+397	A(5) <- C=1	3	0	message
n+408	n+398	A(4) <- C=1	3	0	dans 50-59
n+409	n+399	A(1) <- C=1	3	0	
n+410	n+355	M(21) <- A	3	0	
n+411		erreur de débordement			
n+412		écart date trop important			
n+413		erreur de décodage			
n+414		Ramda (61) = flagMPE = 1			début du test

adresse courante	adresse suivante	commande	fintest	C	
n-1	n	init générateur test	0	0	test CDR self exercising
n	n+2	déclenche test CDR	1	0	
fintest = faux -> n+1	n	NOP	0	0	
fintest = vrai -> n+2	n+3	(DT,DT*) = (1,0)	0	0	test micro-ROM
					On suppose que la dernière
					adresse du test de la micro
n+3	n+4	A <- AUX	0	0	ROM est n+3 test AU
n+4	n+6	B<- AUY et A<- A+B+C	1	0	
fintest = faux -> n+5	n+3	NOP	0	0	
fintest = vrai -> n+6	n+7	C", C' <- générateur test	0	0	test UL
n+7	n+9	C" <- C" opé C'	1	0	
fintest = faux -> n+8	n+6	NOP	0	0	
fintest = vrai -> n+9	n+10	RAM(?) <- RS	0	0	
n+10	n+12	CDR bus et RS <- générateur tes	1	0	test CDRbus et sortie
fintest = faux -> n+11	n+10	NOP	0	0	
fintest = vrai -> n+12	n+13	A <- 0	0	0	fin de test des sorties
n+13	n+14	A <- 255	0	0	
n+14	n+15	RS <- A	0	0	
n+15	n+16	RS <- 0	0	0	
n+16	n+17	RS <- A	0	0	
n+17	n+18	RS <- A	0	0	mot hors code car RS*<-A
n+18	n+19	RS <- RAM(?)	0	0	
n+19	n+20	A <- 0	0	0	test RAM
n+20	n+21	RI <- A	0	0	
n+21	n+22	B <- RAM(RI)	0	0	code
n+22	n+23	RAM(RI) <-B*	0	0	hors code
n+23	n+24	RAM(RI) <-B	0	0	code
n+24	n+25	RAM(RI) <-B*	0	0	hors code
n+25	n+26	A <- A + 1	0	0	
n+26	n+28	C <- A(6)	0	1	
C = 0 -> n+27	n+21	RI <- A	0	0	
C = 1 -> n+28	n+29	A <- 0	0	0	
n+29	n+30	RI <- A	0	0	
n+30	n+31	B <- RAM(RI)	0	0	hors code
n+31	n+32	RAM(RI) <-B*	0	0	code
n+32	n+33	Bus <- RAM(RI)	0	0	code
n+33	n+34	RAM(RI) <-B	0	0	hors code
n+34	n+35	Bus <- RAM(RI)	0	0	hors code
n+35	n+36	A <- A + 1	0	0	
n+36	n+38	C <- A(6)	0	1	
C = 0 -> n+37	n+30	RI <- A	0	0	
C = 1 -> n+38	n+39	A <- A - 1 ou A+FF	0	0	
n+39	n+40	RI <- A	0	0	
n+40	n+41	B <- RAM(RI)	0	0	hors code
n+41	n+42	RAM(RI) <-B*	0	0	code
n+42	n+43	RAM(RI) <-B	0	0	hors code
n+43	n+44	RAM(RI) <-B*	0	0	code
n+44	n+46	A <- A - 1 ou A <- A +FF	0	1	
C = 0 -> n+45	n+47	A(7) <- C=0	0	0	A = 127
C = 1 -> n+66	n+40	RI <- A	0	0	
n+47	n+48	A(6) <- C=0	0	0	A = 63
n+48	n+49	RI <- A	0	0	
n+49	n+50	B <- RAM(RI)	0	0	code
n+50	n+51	RAM(RI) <-B*	0	0	hors code
n+51	n+52	Bus <- RAM(RI)	0	0	hors code
n+52	n+53	RAM(RI) <-B	0	0	code
n+53	n+54	Bus <- RAM(RI)	0	0	code

n+54	n+56	A <- A - 1 ou A <- A +FF	0	1	
C = 0 -> n+55	n+57	COl <- 0	0	0	
C = 1 -> n+56	n+49	RI <- A	0	0	
n+57	n+58	COh <- 0	0	0	test ROM externe
n+58	n+59	RI<- ROM	0	0	
n+59	n+60	COl <- COl + 1	0	0	
n+60	n+61	COh <- COh + C"	0	0	
n+61	n+62	A <- COh	0	0	
n+62	n+64	C <- A(4)	0	1	
C = 0 -> n+63	n+60	COl <- COl + 1	0	0	
C = 1 -> n+64	n+65	A <- 0	0	0	
n+65	n+66	A(?) <- C avec C = 1	0	0	
n+66	n+67	A(??) <- C avec C = 1	0	0	
n+67	n+68	déclencher analyse signature	0	0	
n+68	n+ 70	A <- A - 1 ou A <- A + FF	0	1	
C = 0 -> n+69	n+71	B <- RAM(COl)	0	0	
C = 1 -> n+70	n+67	NOP	0	0	
n+71	n+72	COl <- B	0	0	restauration données
n+72	n+74	B <- RAM(COh)	0	0	
n+73	n+69	COh <- B	0	0	
n+74	n+75	A <- RAM(C)	0	0	
n+75	n+76	C <-A	0	0	
n+76	n+77	A <- RAM(A)	0	0	
n+77	n+79	NOP lecture compteur	0	com=1	débordement compteur
com = 0 -> n+78	n+80	NOP	0	0	
com = 1 -> n+79	???	erreur	0	0	
n+80	n+82	NOP lecture compteur	0	com=1	
com = 0 -> n+81	n+80	NOP incrémentation compteur	0	0	
com = 1 -> n+82	zz =2	compteur <- 0	0	0	test du RI
x	x+1	A <- 0	0	0	initialisation
x+1	x+2	RI <- A	0	0	
x+2	x+3	RAM <- 0	0	0	
x+3	x+4	A <- A + 1	0	0	
x+4	x+6	C <- A(6)	0	1	
C = 0 -> x+5	x+2	RI <- A	0	0	
C = 1 -> x+6	x+7	A <- 0	0	0	
x+7	x+8	RI <- 0	0	0	
x+8	x+9	B <- RAM(0) = 0	0	0	
x+9	n-1	A <- 0	0	0	

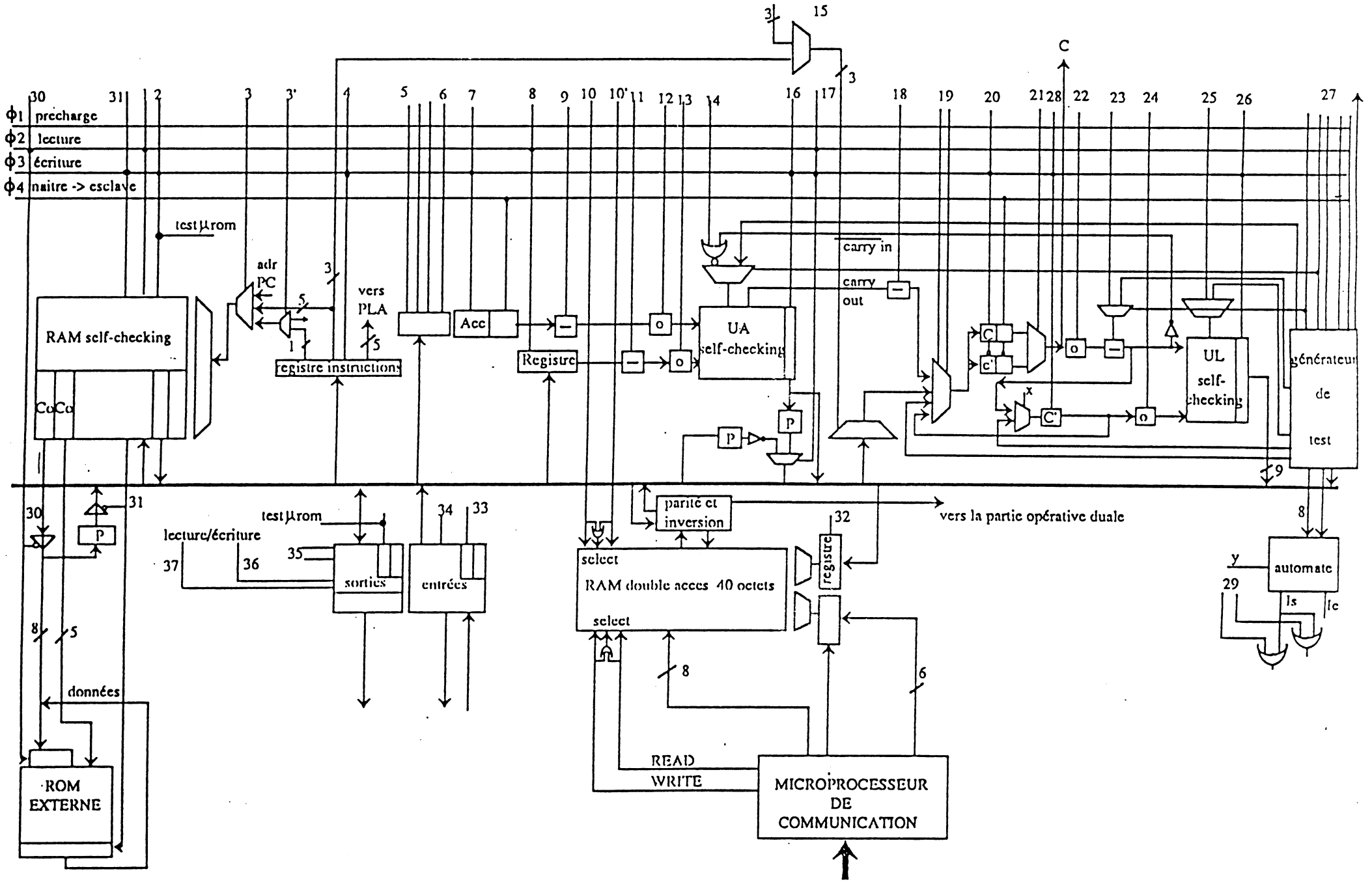
adresse courante	adresse suivante	commande	testRI	testC
0	1	COl <- 0	0	0
1	2	COh <- 0	0	0
2	3	RI <- ROM	1	0
31	4	ADD	0	0
32	4	SUB	0	0
33	4	LOAD	0	0
34	4	STORE	0	0
35	4	ET	0	0
36	4	OU	0	0
37	4	NON	0	0
38	4	PUSH	0	0
39	4	PULL	0	0
310	4	SET	0	0
311	4	RESET	0	0
312	3121	R si C =1 COl <- COl +	0	0
313	3131	R si C = 0 COl <- COl +	0	0
314	3141	BR COl <- COl + 1	0	0
315	3151	VI COl <- COl + 1	0	0
316	x	passiver NOP	0	0
317	z	STOP NOP	0	0
4	5	COl <- COl + 1	0	0
5	2	COh <- COh + C"	0	0
3121	3122	COh <- COh + C"	0	0
3122	3123	RAM <- ROM	0	0
3123	3124	COl <- COl + 1	0	0
3124	3125	COh <- COh + C"	0	0
3125	3127	RAM <- ROM	0	1
C = 0 -> 3126	5	COl <- COl + 1	0	0
C = 1 -> 3127	3148	RegB <- RAM	0	0
3131	3132	COh <- COh + C"	0	0
3132	3132	RAM <- ROM	0	0
3133	3133	COl <- COl + 1	0	0
3134	3134	COh <- COh + C"	0	0
3135	3137	RAM <- ROM	0	1
C = 0 -> 3136	3146	NOP	0	0
C = 1 -> 3137	5	COl <- COl + 1	0	0
3141	3142	COh <- COh + C"	0	0
3142	3143	RAM <- ROM	0	0
3143	3144	COl <- COl + 1	0	0
3144	3145	COh <- COh + C"	0	0
3145	3146	RAM <- ROM	0	0
3146	3147	RegB <- RAM	0	0
3147	3148	COl <- RegB	0	0
3148	3149	RegB <- RAM	0	0
3149	2	COh <- RegB	0	0
3151	3152	COh <- COh + C"	0	0
3152	4	A <- ROM	0	0

ANNEXE II

L'annexe II présente une description générale d'une partie opérative du circuit et une liste des commandes qui permettent de la faire fonctionner. Cette liste donne la largeur du registre de commande. Les contrôleurs double-rail n'apparaissent pas sur ce schéma.

1 lecture RAM
1 Ecriture RAM
1 sélection des adresses venant soit de la PC soit de RI
1 sélection du 6ième bit de l'adresse (test)
1 lecture RI
3 commandes de chargement d'un bit parmi 8
1 chargement de tous les bits
1 ecriture dans l'accumulateur A
1 écriture dans le registre B
1 inversion de la sortie de l'accumulateur
1 lecture de la RAM double-accès
1 écriture dans la RAM double-accès
1 inversion de la sortie du registre B (test)
1 mise à 0 de la sortie du registre B
1 mise à 0 de la sortie de l'accumulateur
1 mise à 0 de la retenue entrante
1 sélection de l'adresse du bit à charger dans C
1 validation des sorties de l'unité arithmétique
1 sélection de la parité (test)
1 inversion de la retenue sortante
2 sélection du bit parmi 4 à charger dans C
1 chargement de C ou C"
1 sélection entre C ou C"
1 mise à 0 de la sortie de C
1 inversion de la sortie de C
1 mise à 0 de la sortie de C'
1 sélection de l'opération logique (ET ou OU)
1 validation du résultat de l'unité logique
5 commandes de test du générateur (testCDR, testUL, testUAX, testUAY, Init)
1 chargement de C'
2 commandes modifiant les sorties de l'automate indicant la nature des mots (test)
1 chargement des adresses dans la ROM externe
1 lecture des données externe
1 chargement du registre d'adresse de la RAM double-accès
3 lecture des 3 registres d'entrée
1 déclenchement du test des entrées
2 test de S9 et test des sorties
1 écriture dans le registre de sortie
1 lecture du registre des sorties

A ces 50 commandes il faut ajouter les commandes de déclenchement et de lecture de l'analyse de signature ainsi que les commandes de déclenchement de l'automate de test de la partie contrôle, soit un total de 54 commandes dont 17 pour le test. Le registre de commande dispose aussi de 6 bits d'adresse avec leur parité.


φ1 precharge
φ2 lecture
φ3 écriture
φ4 maitre -> esclave
testμrom
RAM self-checking
CoCo
adr PC
registre instructions
vers PLA
Acc
Registre
UA self-checking
carry in
carry out
UL self-checking
générateur de test
P
C
C'
c'
x
parité et inversion
vers la partie opérative duale
testμrom
lecture/écriture
sorties
entrées
select
RAM double acces 40 octets
registre
automate
y
ls
le
données
ROM EXTERNE
READ
WRITE
MICROPROCESSEUR DE COMMUNICATION

ANNEXE III

Le test des circuits Self-Checking à logique CMOS dynamique.

Les pannes latentes provoquant des accumulations de pannes dans les circuits Self-Checking peuvent annuler la propriété Fault-Secure.

Pour éviter ces pannes on suppose que les pannes surviennent une par une, et entre l'occurrence de 2 pannes, il s'écoule un temps suffisamment long pour que le circuit reçoive pendant le fonctionnement normal tous les vecteurs du code d'entrée (hypothèse H2). Cette hypothèse est simplifiée lorsqu'on considère un ensemble particulier de pannes (e. g. collage), auquel cas le circuit doit recevoir des vecteurs de test assurant la couverture de panne de collage. Aucune restriction quand à l'ordre d'application des vecteurs de test n'est considérée dans ce cas.

Par contre, la détection de certaines pannes affectant des circuits CMOS demande l'application de séquences de couple de vecteurs.

On doit alors examiner si des restrictions sont à considérer, lorsque l'on applique des séquences de test à des circuits Self-Checking implémentés en logique CMOS à précharge.

la figure 1 a) présente la structure d'une porte CMOS à précharge [WES 85]. De telles portes ont une structure très différente d'une logique FCMOS ou CMOS hybride (figure 1 b)). En effet, la testabilité de telles portes est différente. Un MOS stuck-open, par exemple, dans le réseau n de la figure 1 b) demanderait l'application de 2 vecteurs de test (t1, t2) pour être détecté [WAD 78] [DAR 89]. Quand le vecteur t2 détectant la panne s-open est appliqué, la sortie de la porte est en état de haute impédance (dû à la panne s-open) conservant ainsi sa valeur initiale. Si le vecteur t1 connecte la sortie de la porte au Vss (ce qui est possible puisque le réseau n peut connecter la sortie au Vss à travers des chemins ne passant pas par le MOS s-open), la sortie de la porte est maintenue à la valeur 0 lorsque t2 est appliqué, ce qui correspond à une réponse correcte. Ce qu'il faut faire alors, c'est initialiser la sortie de la porte à 1 par un vecteur t1.

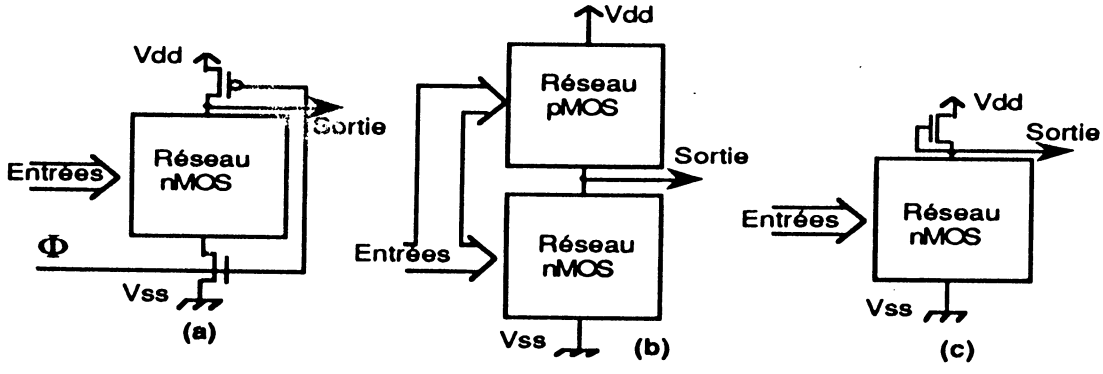


Figure 1 : a) Porte CMOS dynamique, b) FCMOS et CMOS hybride, c) nMOS .

Par opposition, pour une porte implémentée sous forme domino la sortie de la porte est systématiquement préchargée à 0 par t1, durant la phase de précharge, pour que t2 détecte la panne

s-open indépendamment de la séquence précédente.

Une autre différence concerne la testabilité de pannes s-on. Dans le cas du réseau n de la figure 1 b), une panne s-on peut donner des niveaux intermédiaires, puisque quand le vecteur de test est appliqué, la porte se transforme en un diviseur de tension connectant simultanément la sortie de la porte à Vdd et à Vss à travers respectivement le réseau p et le réseau n. Ces pannes ne sont donc pas détectables par des techniques de test à collage logique.

Par contre, quand le vecteur de test est appliqué, une panne s-on dans le réseau n de la figure 1 a) donnera en sortie le niveau logique 0, puisque durant la phase d'évaluation le transistor MOS p est déconnecté.

Par conséquent, la porte de la figure 1 a), manifeste les mêmes comportements de testabilité des pannes de s-on et s-open de transistors MOS dans le réseau n, que l'implémentation nMOS de la figure 1 c).

Un cas particulier est celui du transistor MOS p de précharge. Pour détecter les pannes de s-open de ce transistor, on doit appliquer sur les entrées de la porte, un vecteur d'entrée pour lequel la sortie prend la valeur "1". Quand ce vecteur est appliqué, la sortie de la porte est en état de haute impédance (le comportement est similaire au cas d'une implémentation nMOS pour laquelle la connexion avec Vdd est coupée).

Il résulte que les circuits Self-Checking implémentés en logique CMOS à précharge (et même pour une logique nMOS) demande à être activés régulièrement par 2 séquences de test. Mais la situation est différente de celle des logiques FCMOS et CMOS hybride.

Dans de telles portes, pour un transistor p s-open (resp. transistor n s-open), plusieurs chemins peuvent exister connectant la sortie de la porte à Vdd (resp. Vss) et ne passant pas par le MOS s-open de façon à ce que juste avant l'application du vecteur de test détectant la panne s-open, la sortie de la porte peut être initialisée à la valeur correcte.

D'autre part, dans le cas d'une porte dynamique, la sortie de la porte est déconnectée indéfiniment de Vdd. Il en résulte que les hypothèses standards utilisées pour des circuits Self-Checking à logique nMOS sont aussi valables et suffisantes pour des circuits Self-Checking à logique CMOS préchargée.

En fait, il est certain que quelque temps après l'occurrence d'une panne s-open, un vecteur pour lequel la sortie de la porte prend la valeur 0 survient (sinon le circuit Self-Checking n'aurait pas été activé pour un collage à 1 de la sortie). Après un tel vecteur, la sortie reste à 0 jusqu'à l'apparition d'un vecteur détectant le collage à 0 de la sortie.

Ce vecteur détectera donc la panne s-open et son occurrence est garantie si une génération standard de vecteurs de test est effectuée. Les hypothèses standards concernant la génération de vecteurs de test sont donc suffisantes pour activer des circuits Self-Checking implémentés en logique CMOS à précharge (comme pour une logique nMOS).

Un autre cas concerne les pannes s-on affectant un transistor de précharge p ou n. Ces pannes induisent des valeurs de sortie à un niveau indéterminé, leur détection n'est pas garantie.

Si l'on veut détecter ces pannes, il faut que leur occurrence produise des valeurs logiques à la

sortie de la porte.

Si la sortie d'une porte prend la valeur logique 0 (resp. 1) quand les deux réseaux n et p conduisent en même temps, la porte est dite n-dominante (resp. p-dominante) [OKL 84].

Ceci est réalisable en dimensionnent de manière adéquate les transistors, comme dans le cas d'une logique pseudo-nMOS (figure 2).

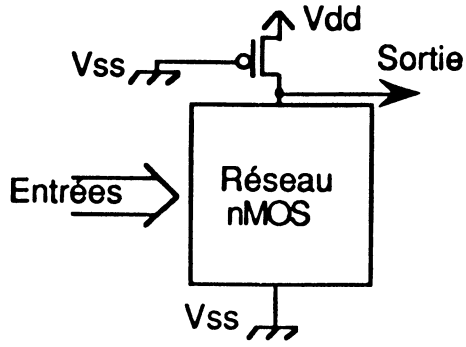


Figure 2 : Porte pseudo-nMOS.

Si l'on considère que les portes sont p-dominantes. Il suffit d'augmenter la largeur de canal du transistor de précharge p de la porte en figure 1 a). Une conception n-dominante de cette porte demanderait l'augmentation de la largeur de canal des transistors n.

Une conception p-dominante est préférable à une n-dominante, en effet pour une porte n-dominante on doit tailler plusieurs transistors n, et la détectabilité d'un s-on du transistor p de précharge dépendrait du vecteur d'entrée durant la phase de précharge.

D'autre part, les courts-circuits entre sorties de portes dynamiques sont détaillés dans [NIC 87], ils ont les mêmes conséquences que ceux affectant les portes nMOS.

ANNEXE IV

Testabilité de contrôleurs double-rail CVSL dynamiques

L'étude présentée en annexe III peut être étendue à une logique CMOS domino représentée en figure 3 [WES 85].

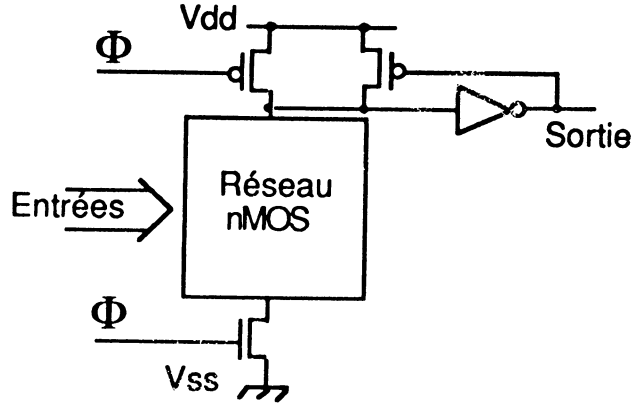


Figure 3 : Logique CMOS domino.

En comparaison aux portes CMOS dynamiques (figure 1 a)) on a 3 transistors MOS supplémentaires dans une logique domino.

Un transistor p de rebouclage et un inverseur CMOS avec un transistor n et un transistor p. Comme en annexe III on considère que la porte est p-dominante.

L'inverseur peut tout aussi bien être p-dominant ou n-dominant, mais une implémentation sous forme n-dominante est plus efficace, elle sera donc adoptée.

Sous cette condition les pannes s-on des 3 MOS supplémentaires sont testables par un test à modèle de collage logique (pannes s-on du transistor p de rebouclage et le transistor n de l'inverseur) ou ne modifie pas les valeurs des sorties (pannes s-on du transistor p de l'inverseur).

Une panne s-open sur le transistor p de rebouclage ne modifie pas les valeurs de sortie.

Dans le cas de panne de s-open du transistor p de l'inverseur, la sortie est déconnectée en permanence du Vdd, et une panne s-open du transistor n de l'inverseur déconnecterait en permanence la sortie du Vss.

En considérant les mêmes arguments que ceux présentés en annexe III à propos de la testabilité des pannes s-open du MOS p de précharge, les hypothèses standard d'application de vecteurs de test à des circuits Self-Checking assurent la testabilité tantôt d'un collage à 1, tantôt d'un collage à 0 de la sortie. Ce qui signifie que la sortie change à certains moments de 1 à 0. Ce qui permet la détection de la panne s-open du transistor n de l'inverseur qui empêche une transition de la sortie de 1 à 0. De la même façon, une panne s-open du transistor p de l'inverseur est détectable.

En conclusion, la porte de la figure 3 est suffisamment activée si elle reçoit régulièrement un ensemble de vecteurs de test activant complètement la même porte implémentée en logique nMOS (i.

e. la porte nMOS correspondante a le même réseau de décharge que le réseau n de la porte de la figure 3).

Il est facile de voir que ces résultats peuvent directement être appliqués au contrôleur double-rail de la figure 4.

Plusieurs séquences de test existent ayant chacune 4 vecteurs de test et pouvant chacune activer l'implémentation nMOS du contrôleur. Elles peuvent être utilisées pour activer une implémentation CVSL de ce contrôleur.

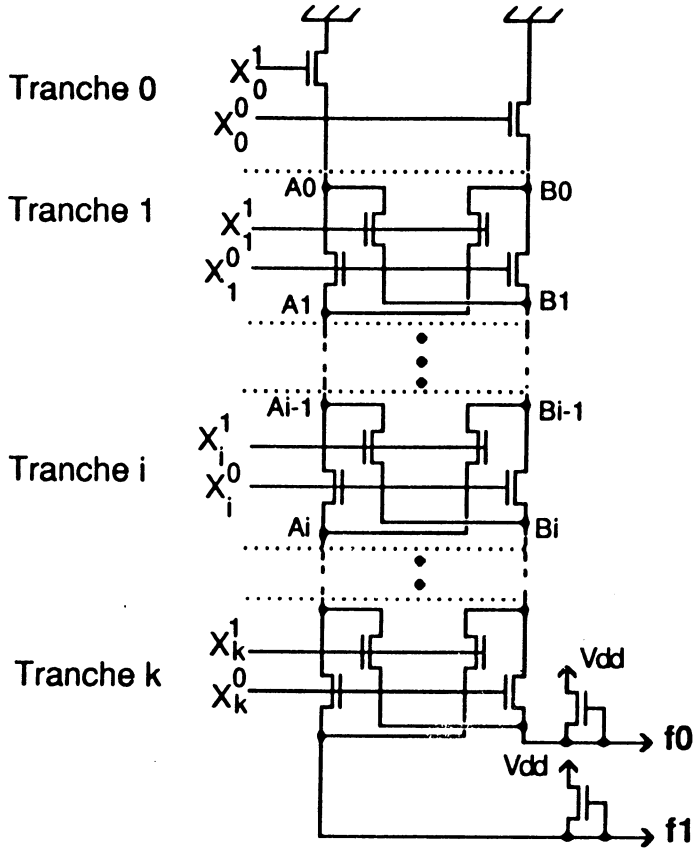


Figure 4 : Implémentation CVSL du contrôleur double-rail à k variables.

Remarque 1 : Si en figure 4, les deux noeuds Ai et Bi de la tranche i sont connectés à (resp. déconnectés de) Vss, les sorties f0 f1 prennent toutes les deux la valeur 0 (resp. 1).

A partir de cette remarque, on déduit que si une panne provoque la connexion ou la déconnexion du Vss des deux noeuds de sortie d'une tranche i, la panne est détectée.

L'état d'un noeud connecté au Vss sera noté 0 et l'état d'un noeud déconnecté du Vss sera noté H.

Remarque 2 : Les pannes simples de collage, s-on, et s-open dans la tranche 0 de la figure 4 sont détectées si l'on applique aux entrées X^0_0 X^1_0 les vecteurs (1,0) et (0,1).

Remarque 3 : Les pannes simples de collage, s-on, et s-open dans la tranche i sont détectées si l'on applique sur les entrées (X^0_0 X^1_0), (X^0_1 X^1_1), ..., (X^0_i X^1_i) 4 vecteurs (V1, V2, V3, V4) provoquant sur X^0_i, X^1_i, A_{i-1}, B_{i-1} 4 états donnés en table I.

	Ai-1	Bi-1	Xi^0	Xi^1	Ai	Bi
V1	0	H	0	1	H	0
V2	0	H	1	0	0	H
V3	H	0	0	1	0	H
V4	H	0	1	0	H	0

Table I : Les états permettant le test de la tranche i.

La preuve de cette remarque est évidente.

Il est facile de voir qu'il existe un seul ensemble de 4 vecteurs de test appliqués sur X^0_0, X^1_0, X^0_1, X^1_1 générant les 4 états de la table I et testant ainsi la tranche 1. Ces mots de code sont {0101, 0110, 1001, 1010}.

Si on considère les sorties A_i, B_i de la tranche i de la table I, on trouve forcément que pour tester la tranche i+1 il y a 4 façons de combiner les vecteurs V_1, V_2, V_3, V_4 avec des mots du code $X^0_{i+1}\ X^1_{i+1}$=01 et $X^0_{i+1}\ X^1_{i+1}$=10. Ces combinaisons sont (V_1 01, V_2 01, V_3 10, V_4 10), (V_1 01, V_2 10, V_3 01, V_4 10), (V_1 10, V_2 01, V_3 10, V_4 01) et (V_1 10, V_2 10, V_3 01, V_4 01). Ces combinaisons donnent en Ai, Bi, X^0_{i+1}, X^1_{i+1} les 4 ensembles de 4 vecteurs suivants: (H001, 0H01, 0H10, H010), (H001, 0H10, 0H01, H010), (H010,0H01, 0H10, H001), (H010, 0H10, 0H01, H001). Chacun de ces ensembles assure la testabilité de la tranche i+1. De chacun des ensembles de vecteurs de test de la tranche i on obtient 4 ensembles de vecteurs de test testant la tranche i+1. Comme l'on a un seul ensemble de vecteurs de test de 4 mots du code testant les tranches 0 et 1, pour un contrôleur à n variables on aura 4^{n-2} différents ensembles de test de 4 mots du code, chacun de ces ensembles assure le test du contrôleur.

La figure 5 montre un contrôleur double-rail à 2 couches cascadant plusieurs petits contrôleurs. Si on considère un groupe composé de l ensembles de vecteurs de test, chacun de ces ensembles contenant 4 mots du code du contrôleur double-rail de la première couche. On a exactement $(4^{n_1}-2)x(4^{n_2}-2)x$ .... $x(4^{n_l}-2)$ de tels groupes.

Par une analyse similaire au cas d'une réalisation sur une couche, on peut montrer que les 4 ensembles de vecteurs du code d'un tel groupe peut être combiné de 4^{2l-2} différentes façons pour donner des ensembles pour le contrôleur double-rail à l variables. Par conséquent on a $(4^{n_1}-2)$ x $(4^{n_2}-2)$ x .... x $(4^{n_l}-2)$ x 4^{2l-2} $=4^{(n_1+n_2+....+n_l-2)}$ ensembles de test de 4 mots du code, chacun assurant le test de tout le contrôleur.

Ce résultat peut être étendu à tout nombre de couches, de façon à ce que pour une réalisation d'un contrôleur double-rail à k variables on a 4^{k-2} ensembles de 4 mots du code de test. On peut conclure par le fait que le contrôleur est testable par un grand nombre d'ensembles de test de 4 mots du code impliquant une très bonne testabilité.

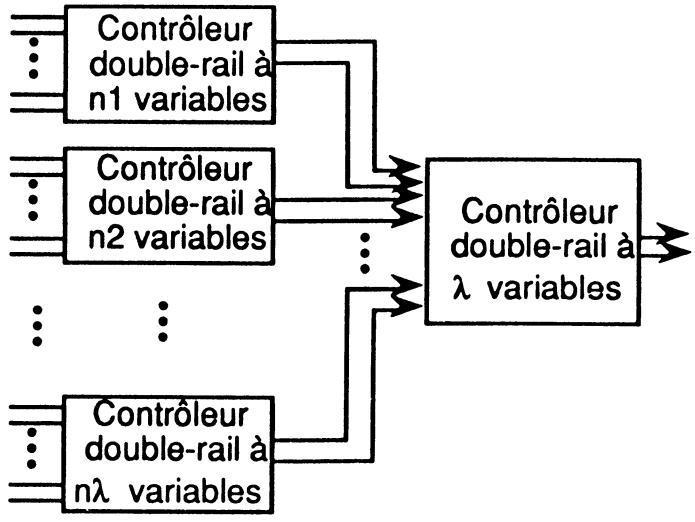


Figure 5 : Contrôleurs double-rail en cascade.

ANNEXE V

PLAs implémentés en logique CMOS dynamique

[MAK 82] présente des implémentations nMOS de PLAs ayant leurs sorties codées en codes non ordonnés. Ces PLAs sont SFS vis à vis de pannes simples de collage, de transistor MOS de croisement manquant ou parasite, et de courts-circuits.D'autre part dans [NIC 86], on présente des règles de conception de PLAs assurant la propriété SFS pour les pannes ci-dessus, plus des pannes s-on et s-open. Ces règles concernent une technologie nMOS, mais leur extension à une technologie CMOS à précharge est directe.Toute panne simple, de l'ensemble décrit ci-dessus, survenant dans un PLA produit une erreur simple sur une ligne d'entrée, ou une ligne de monôme, ou une ligne de sortie. Dans un PLA, tous les chemins reliant une ligne (ligne d'entrée, de monôme, ou de sortie) et une ligne de sortie a la même parité d'inversion. Les erreurs simples précédentes sont propagées en erreurs unidirectionnelles aux sorties du PLA.Une panne de collage du signal de précharge dans la matrice ET (resp. matrice OU) produit des erreurs multiples sur les lignes de monôme (rep. les lignes de sortie).On peut avoir les quatre situations suivantes :

Collage à 0 du signal de précharge dans la matrice ET : Cette panne met à 1 toutes les lignes de monôme, toutes les erreurs sont du type $0\rightarrow1$. Ces erreurs unidirectionnelles sur les lignes de monôme sont propagées aux sorties du PLA en erreurs unidirectionnelles du type $1\rightarrow0$. Ceci est dû au fait que la parité d'inversion entre une ligne de monôme et une ligne de sortie est impaire.

Collage à 1 du signal de précharge dans la matrice OU : Cette panne empêche la précharge des lignes de monôme du PLA, il en résulte que toutes les lignes de monôme ont une erreur du type $1\rightarrow0$. Il en résulte, comme précédemment, que les erreurs sont propagées aux sorties du PLA en erreurs unidirectionnelles du type $0\rightarrow1$.

Collage à 0 du signal de précharge dans la matrice ET : Cette panne engendre des erreurs unidirectionnelles du type $0\rightarrow1$ à toutes les lignes de sortie du PLA.

Collage à 1 du signal de précharge dans la matrice OU : Cette panne engendre des erreurs unidirectionnelles du type $1\rightarrow0$ à toutes les lignes de sortie du PLA.
Le PLA est ainsi muni de la propriété Fault-Secure. En annexe III on a étudié la détectabilité de pannes dans les portes CMOS dynamiques et on a vu qu'elle était la même que pour une logique nMOS. La propriété SFS est donc assurée pour un PLA CMOS dynamique de la même façon qu'elle l'est pour un PLA nMOS.

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

INPG

Grenoble, le 28 Septembre 1990

DÉPARTEMENT DES ÉTUDES DOCTORALES

Affaire suivie par
Tél : 76.57.

N/Réf. :

Objet :

AUTORISATION de SOUTENANCE

Vu les dispositions de l'arrêté du 23 Novembre 1988 relatif aux Etudes Doctorales
Vu les rapports de présentation de :

- Monsieur GREINER
- Monsieur TEIXEIRA

Monsieur CHAUMONTET Gilles

est autorisé(e) à présenter une thèse en soutenance en vue de l'obtention du diplôme de DOCTEUR de l'INSTITUT NATIONAL POLYTECHNIQUE de GRENOBLE, spécialité : "Microélectronique"

Pour le Président
et par délégation
Le Vice-Président

C. GAUBERT

RESUME

Actuellement, toutes les applications critiques mettant en jeu la vie humaine ne peuvent pas être assurées par des systèmes complexes utilisant des circuits intégrés répliqués; il est necessaire d'utiliser des composants discrets de sécurité intrinsèque, d'un encombrement et d'un coût prohibitifs.

Pour relever ce défi, le micro-contrôleur MAPS qui doit gérer la signalisation ferroviaire, bénéficie de l'intégration d'un circuit logique autotestable, en-ligne (duplication duale + parité) et hors-ligne, suivant le principe de la technique UBIST. Le MAPS dispose aussi d'une interface de sortie apte à produire des signaux de commande en fréquence, soit sûrs soit corrects. Il dispose également d'une interface d'entrée capable de n'accepter des signaux externes qu'après les avoir rendus sûrs ou corrects. Ces deux interfaces intégrées pour la première fois, sur la même puce que le circuit autotestable, sont "strongly fail-safe". Seules les communications avec l'extérieur se font par échange de messages fortement codés sans qu'aucun matériel ne soit rajouté.

En conséquence, l'étude que l'on présente permet d'apporter une nouvelle démarche de conception des systèmes hautement critiques, tout en assurant un degré de sécurité nettement plus élevé (détection de toutes les pannes triples) que celui donné par les systèmes actuels, et ceci pour un volume et un coût plus faibles.

MOTS-CLEF

circuits auto-contrôlables - test en-ligne - test hors-ligne - UBIST - micro-contrôleur - signalisation ferroviaire - strongly fail-safe - hypothèse de panne - duplication - sorties fréquence