M2 COMASIC
Examen de Sûreté de Fonctionnement 29/01/2018

Durée de l'examen 2h30.

Consignes particulières : les documents sont autorisés. Les téléphones portables doivent être rangés. Le présent examen est noté sur 20 pts. Le barème est détaillé devant chaque question à titre indicatif de l'importance de chaque question.

I Principes de la sûreté de fonctionnement (5,5 pts)

Question 1) (2 pt) Donnez un exemple où un même événement (ou état du système) peut être, en fonction du périmètre dans lequel l'événement est étudié, une erreur, une faute, ou une défaillance. (maximum 1/2 page).

Réponse : prenons le cas d'une application réalisant une compression d'une vidéo afin de la diffuser. L'application repose sur des fonctions f1 et f2. La fonction f1 réalise un premier calcul sur les données d'entrée D et produit un résultat de même taille que D. Le résultat f1(D) est placé dans une file dont le contenu est utilisé comme paramètre de f2. Le résultat de l'application de f2 au contenu de la file correspond à la vidéo compressée au format souhaité.

Considérons l'événement où f1 place dans la file une sortie tronquée à la moitié de la taille attendue. C'est une défaillance de f1, une erreur pour le système (car la file contient une donnée tronquée), et une faute pour f2 car la donnée d'entrée étant corrompue, elle a le potentiel de déclencher une défaillance.

Question 2) (1,5 pts) Classes de défaillances

Nous considérons quatre classes de défaillance pour les systèmes en réseau :

- Crash (la machine n'émet plus aucun message)
- Omission (certaines messages peuvent ne pas être envoyés)
- Commission (la machine réémet certains messages)
- Byzantin (la machine peut avoir n'importe quel comportement imaginable –sur le réseau)

a) (0,5 pt) Décrivez les relations d'inclusion de ces différents ensembles de défaillances.

Réponse : Byzantin contient omission et commission qui sont disjoint, omission contient crash

b) (1 pt) Dans le cas d'un contrôleur de vitesse sur une voiture expliquez en quoi la classe de défaillance crash représente une amélioration de la

sécurité-innocuité du composant par rapport à une défaillance byzantine. On supposera que le contrôleur de vitesse communique sa sortie par une variable partagée au contrôle moteur.

Réponse : si le contrôleur de vitesse émet des commandes arbitraires il peut amener le véhicule à une vitesse dangereuse sans que cela soit concrètement détectable. En revanche, un crash du calculateur se traduit par une consigne constante. Ce comportement est d'un point de vue pratique beaucoup moins dangereux qu'une brusque accélération ou décélération.

Question 3) (2 pts) Modes de défaillance et propagation

Nous considérons que le code suivant s'exécute dans un processus Unix standard. Nous supposons que la fonction factorielle(p) doit retourner p!, si p est strictement positif et 1 si nul.

```
int factorielle(int n) {
if (n==0){return 1;}
else {return factorielle(n-1)*n;}
}
//nb : on supposera que le compilateur utilise 4 octets pour le type int
```

a) **(0,5 pts) Dans quelles conditions l'exécution de cette fonction engendre l'arrêt du programme l'exécutant en raison d'un dépassement de la pile d'appel**

Réponse : f(-1) engendre une séquence infinie d'appels récursifs.

b) **(0,5 pts) En supposant que le matériel garantit l'intégrité de l'exécution des programme dont il a la charge, dans quelles circonstances le programme ci-dessus peut il engendrer une valeur négative (une condition suffisante sera accepté si elle est correctement justifiée).**

Réponse : le type int est usuellement représenté sur 4 octets donc de valeur maximal ~ $4\,10^9$ or 10^{13} vaut plus de $6{\sim}10^9$

c) **(1 pt) Modifiez le code ci-dessus afin que les défaillances engendrées par ces fautes puisse être signalés par respectivement les valeurs de retour -1 et -2.**

Réponse : il faut rajouter dès la première ligne un test

if (n<0) {return -1}

if (n>Vmax) {return -2} //ou Vmax vaut par exemple 13

II Tolérance aux fautes (10 pts)

Question 4) (3 pts) Etat de fiabilité / disponibilité

Supposons que pour les composants d'un système, on soit capable de définir pour chaque composant son état : fonctionnel ou défaillant. Il est alors possible de définir les combinaisons d'états des composants du système pour lesquelles ce dernier est fiable (produit un résultat correct), ou bien disponible (produit un résultat). Notez que cette caractérisation dépend du fonctionnement du système en présence de composants défaillants.

a) (1,5 pt) Donnez un exemple d'architecture de réplication et un modèle de défaillance pour lesquels les états de fiabilité et de disponibilité sont les mêmes.

Réponse : une architecture à base de réplication passive, avec des défaillances par crash de ses N répliques, possède la même définition de leur état de fiabilité ou disponibilité : le nombre de ses répliques non défaillante est plus grand que 1. En effet, une réplique sert soit de back up, ou est défaillante, ou produit de manière correcte le résultat attendu. Ainsi tant qu'une réplique peut produire le résultat l'architecture complète est disponible et fiable.

b) (1,5 pt) Donnez un exemple d'architecture de réplication pour laquelle les états de fiabilité et de disponibilité sont différents (malgré le fait que les composants n'aient qu'un seul mode de défaillance).

Réponse : une réponse possible repose sur l'architecture de réplication active à 3 répliques. En supposant que le voteur n'émet les sorties que si le voteur dispose d'au moins deux réponses identiques. Le système est disponible dans les deux cas suivants : au moins deux répliques correctes ou 2 défaillantes en valeur. Or l'état de fiabilité correspond à au moins deux répliques correctes. Une autre possibilité est de considérer que le voteur laisse les données filtrée si au moins 2 réponse identiques, ou si une seule réponse est reçue.

Question 5) (2 pts Réponse vrai / faux mais elles doivent être justifiée

a) Dans le cadre d'un mécanisme de recouvrement avant, il est important de sauvegarder régulièrement l'état d'exécution (afin de le recharger plus tard)

Réponse : faux un recouvrement avant repose sur le fait, après détection, de charger un état correct prédéfini relativement indépendant de l'état historique de l'état erroné.

b) Pour des données représentant 2^N valeurs, il est nécessaire d'avoir au minimum N + 2*r+1 bits pour détecter l'altération de r bits

Réponse : c'est faux
Contre exemple : N=1 r=1 avec 2 bits on peut détecter 1 erreur Soit D l'information à coder, si D=1 le code est 11, si D=0 le code est 00

c) En utilisant la fonction de codage suivante qui prend 4bits B1B2B3B4, et engendre un champ de 8 bits : B1B2B3B4C1C2C3C4 tel que C1= B1 XOR B3, C2= B2 XOR B4, C3=B1 XOR B4, C4=B1 OR B2 OR B3 OR B4.

Peut on garantir que ce codage permet de détecter une faute qui une fois activée altère deux bits quelconques dans le codage

Réponse : Faux prenons B1-4 = 0101 le codage est donc 01010011. Si l'on modifie B2 et C2 en même temps : 01010011 devient 00010111 qui est lui même un mot du code.

d) Il n'est pas nécessaire de munir un composant de mécanismes de détection de défaillance si il est dupliqué et intégré dans une réplication active.

Vrai : la détection se fait par comparaison du résultat fourni seulement.

Question 6) (5 pts) Redondance des données :

Nous supposons que nous souhaitons tolérer des fautes altérant les émissions réalisées sur un réseau. Nous avons à transférer des données de 45 bits pour lesquels on suppose avoir utilisé un codage requérant N+2r+1 =64 (i.e. r=9). On suppose dans un premier tant que les messages sont émis l'un après l'autre.

a) (0,5 pts) Combien de bits altérés consécutifs peuvent ils être tolérés dans le meilleur et pire des cas sur un message complet (on supposera que la distance de Hamming entre deux mots du code est de 19)?

En supposant que chaque message M_i est coupé en 8 morceaux égaux : $M_i = m^0_i, m^1_i, \ldots m^7_i$

Et les messages $M_0, .., M_7$ sont envoyés en respectant l'ordre suivant ;

$m^0_0, m^0_1, m^0_2, m^0_3, m^0_4, m^0_5 m^0_6, m^0_7\ m^1_0, m^1_1, m^1_2, m^1_3, m^1_4, m^1_5 m^1_6, m^1_7\ m^2_0\ m^2_1$ ....

b) (0,5 pts) En supposant ce nouveau mode d'envoi et que les données ne se perdent pas sur le réseau. Combien d'octets consécutifs faut il recevoir au minimum pour obtenir les 8 morceaux d'un message ?

Réponse : 7 * 8 +1 au moins en effet il faut avoir reçut les 7 premier morceaux de M0 ce qui suppose que l'on a reçut tous les M^i_j

c) (2 pts) Combien de bit altérés consécutifs entre le premier bit et le dernier bit du message peuvent être tolérés en supposant le codage parfait (distance de Hamming entre deux mots du code =r) ?

Réponse : 65 bits = 8 octets +1 bit car chaque groupe de 8 bit d'un message est séparé par 56 bits

d) (0,5 pts) En supposant que l'on puisse redéfinir la taille de la décomposition des messages en morceaux et en respectant le principe d'entrelacement décrit pour la question b), quelle est la taille la plus favorable du point de vue de la tolérance à l'altération de bits contigus.

Réponse : 1 bit par morceaux, cela réparti l'impact des bits altérés sur un plus grand nombre de messages.

e) (1,5 pts) Donnez la valeur du nombre maximal de bits pouvant être altérés de manière consécutive durant l'émission d'un message émis comme décrit en c) ?

Réponse : il y a 64 bits dans chaque messages, donc on retrouve un bit d'un message donné tous les 64 bits. 19 bit consécutifs peuvent être tolérés 19 *64 = 1280-64= 1216 bits consécutifs peuvent être tolérés

III Analyse de disponibilité / fiabilité (4,5 pts)

On souhaite modéliser la fiabilité d'un processeur à l'issue du processus de production. Chaque partie du processeur est susceptible d'avoir souffert de problème à la production rendant le composant défectueux.

Question 1) (1 pt) Quel modèle serait le plus adapté pour modéliser la probabilité que le processeur reste fonctionnel sachant que la probabilité de chaque composant d'être défaillant ne dépend que du type de composant produit (et non de la durée du processus de production). Choisissez en justifiant parmi : reliability block diagram, chaîne de markov à temps discret ou continu.

Réponse : Nous faisons l'hypothèse qu'un processeur est réalisé à partir de composants dont une certaine combinaison est nécessaire pour avoir un processeur fonctionnel. Le modèle qui semble le plus approprié est un reliability block diagram ici car le phénomène aléatoire se produit uniquement lors de la production : son effet est permanent. Soit le

composant est produit fonctionnel soit il est produit défectueux. L'état fonctionnel ne change pas au cours du temps, ni des sollicitations du processeur.

Question (0,5 pts) Avec le modèle de votre choix donner la fiabilité d'un processeur composé d'un coeur d'exécution dont la fiabilité de production est q_p, une mémoire de fiabilité q_m et une glue d'interconnexion entre la mémoire et le coeur d'exécution de fiabilité q_g dans le cas où aucune redondance n'est utilisée

Réponse :

Lemma : (qui pour moi était évident) Soit A,B,C trois événements aléatoire

P(not A et not B et not C) = 1- P (A ou B ou C)

= 1- (P(A) +P(B ou C) – P(A et (B ou C)) = 1 – (P(A) +p(B)+p(C) -P(B) P(C) – P(B et A ou A et C))=

1-P(A)-P(B)-p(C) +P(B) P(C) +P(A) P(B) +P(A)*P(C)-P(A)P(C)P(B)

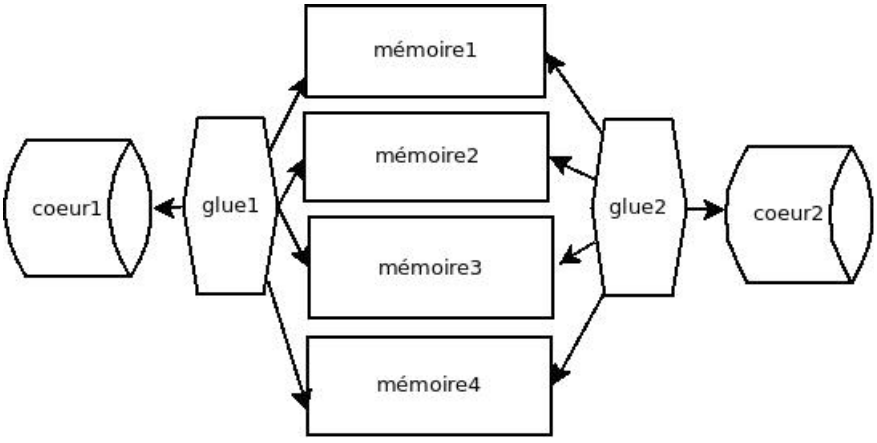
= (1-p(A))*(1-P(B))*(1 -P(C))

Je pensais que le résultat était connu mais même dans le cas contraire ce développement reste rapide à réaliser

du coup R= qm * qc* qg

ps : il n'était pas obligatoire de redémontrer le lemme. Cependant, sa connaissance était a priori considérée comme connue.

Question) (1,5 pts) Même question dans le cas où l'on a retenu l'architecture présentée dans le schéma ci-dessous (si un coeur est connecté à une glue et une mémoire, il est alors possible de réaliser des calculs) :



Réponse : soit xc1 xc2 l'état des coeurs, xg1 et xg2 l'état de la glue, et xm1 ....xm4 les états respectifs de la mémoire :

Décomposons l'état de fiabilité : F1 : traduit le fait qu'au moins un coeurs fonctionnel est connecté aux mémoire :

F1 = ((xc1 et xg1) ou (xc2 et xg2)) en utilisant la formule P(AuB)=P(A)+P(B)-P(AuB) on a

P(F1) = P(xc1 et xg1) + P(xc2 et xg2) – P(xc1 et xc2 et xg1 et xg2)

= 2*qc*qg- $(qc*qg)^2$

F2 correspond à « au moins une mémoire est fonctionnelle) c'est du 1 parmi 4

On applique la formule du cours F2= xm1 ou xm2 ou xm3 ou xm4 et P(F2) = 1- $(1-qm)^4$

Ce qui donne la formule suivante : R= $(2(qg*qc)-(qc*qg)^2) * (1-(1-qm)^4)$

A partir de maintenant nous supposons que les erreurs de fabrication peuvent donner lieu à une modèle de défaillance de type byzantin. Nous supposons que chaque accés au composant mémoire sera traité par un contrôleur mémoire implémentant une architecture de type réplication active à 3 répliques.

Question 2) (1,5 pts) Dans cette architecture, on essaie de déterminer pour quelle fiabilité minimale d'un bloc mémoire peut on utiliser la réplication active à 3 répliques (supposons le voteur parfait) et garantir un accroissement de la fiabilité de 12,5 % (i.e. 1/8). On supposera que les composants ont des probabilité identique d'être fiable égales à *q pour chaque accès mémoire.* De plus le système est fiable si l'on peut garantir que 2 composants mémoires au moins le sont.

Rappel mathématique $(X-c)^2 = X-2cX+c^2$

Réponse :

Analyse de la question : l'accroissement de la fiabilité est à considérer entre l'utilisation d'une seule réplique et l'utilisation d'une architecture TMR.

Soit R_1 la fiabilité pour une réplique $R_1 = qm$ d'après le sujet

Soit R_3 la fiabilité de TMR : $R_3 = q^3 + 3\ q^2*(1-q)$

On souhaite un accroissement de 1/8 donc $R_3 / R_1 = 9/8$

et donc calculons le cas où $R_3/R_1 = 9/8$

Cela donne $q^2+3\ q(1-q)= 9/8$

équivalent à $-2q^2 +3*q-9/8=0$

$q^2 -3/2\ q+ 9/16=0 \sim (q-3/4)^2=0$

La valeur demandée est donc 3/4.