

QEMU

pour le développement embarqué

Thomas Monjalon
thomas.monjalon@openwide.fr

mars 2011

- Description
- Avantages
- Mise en place
- Exemples d'utilisation
 - Tests automatisés (anti-régression)
 - Debug
 - Couverture de test

- 2003 création par Fabrice Bellard
 - 2003 architectures x86, ARM, SPARC
 - 2004 architecture PPC
 - 2005 architecture MIPS
 - 2006 architecture SH4
 - 2007 architectures M68K, Alpha, CRIS
- 2008 rachat de Qumranet (KVM, Spice) par RedHat
- 2008 remplacement du cœur dyngen par TCG
 - 2009 architecture microblaze
 - 2010 interface QMP
 - 2011 architecture LM32
- Depuis, RedHat anime la majorité des contributions.
- IBM s'investit également
 - dont Anthony Liguori, mainteneur

- Émulation de périphériques + hyperviseur KVM = virtualisation
 - exemple : partitionnement de serveur
- Émulation processeur avec TCG (Tiny Code Generator) = user mode
 - exemple : développement d'application embarquée
- Émulation complète avec TCG = **simulation**
 - exemple : développement de système embarqué

Simuler du matériel en langage C

- Environnement entièrement logiciel ou partiellement = cosimulation
- Facilités en environnement logiciel
 - accès au fonctionnement interne du processeur
 - interaction avec des outils logiciels

- Matériel non disponible : obsolète, trop cher, non transportable, en développement, SDK...
- Debug système non intrusif (*)
- Tests automatisés avec stimuli extérieurs (*)
- Couverture de test non intrusive (*)
- Arrêt/reprise de session de mise au point

Dans tous les cas, les contraintes, incidents et doutes matériels n'existent plus.

(*) détaillé plus loin

- Exécution à rebours
- Debug applicatif sans gdbserver
 - pas « d'OS awareness »

1 produit en développement

⇒ 1 simulateur à mettre en place

- Existant ?
 - cartes : environ 30 préconfigurations
 - processeurs : x86, ARM, MIPS, SH4, PPC, etc...
 - périphériques : variés
- Suffisamment complet / proche de la réalité ?
 - approche feignante : on ne simule que ce dont le système a besoin

- Besoin ?
- Classification de l'effort restant :
 - 0) le composant est déjà modélisé
 - 1) il existe un modèle proche que l'on adapte
 - 2) le composant n'est pas très important ⇒ "fake"
(exemple : détection obligatoire)
 - 3) le composant matériel peut-être interfacé
⇒ exécution mixte simulateur+réal
 - 4) le composant est nouveau ou n'a jamais été modélisé

une fois mis en place,

3 exemples d'utilisations

- 1) tests automatisés
- 2) debug
- 3) couverture de test

- Bus de communication (série, ethernet, USB)

le "**tout logiciel**" ouvre des possibilités :

- Commandes d'allumage/extinction
- QMP = système de communication bidirectionnelle au format JSON
 - influencer sur l'exécution (hotplug, boutons, défauts matériels...)
 - statut et événements (erreurs I/O, watchdog...)

- Écriture de scripts de tests au cours du développement
- Utilisables en intégration continue car facilement automatisables
 - tests à chaque commit

- Régression avérée
 - puis écriture du test correspondant
- Heureusement, le programme est versionné avec Git ou un gestionnaire compatible (svn, hg)
- « **git bisect run** <script> » trouve le commit fautif automatiquement
 - le script compile, exécute QEMU et détecte l'erreur
 - codes de retour du script :
 - 0 = OK
 - 125 = non testable
 - 1-124, 126-127 = régression
 - 128-255 = abandon

- 1) tests automatisés
- 2) debug
- 3) couverture de test

- Environnement réel : debug avec sonde JTAG
- Environnement simulé : stub gdb dans Qemu
- GDB ouvre un seul binaire
 - bootloader
 - noyau
 - init
 - application bare board
- Application userland nécessite « OS awareness »
 - dans ce cas, gdbserver est nécessaire
- N'importe quel front-end GDB est utilisable

- Binaire non instrumenté pour le simulateur
- Binaire avec infos de debug (gcc -g3) pour GDB
- Serveur pour GDB accessible par l'option « -gdb »
 - Raccourci « -Ss » = attente connexion sur port 1234
- Connexion GDB = commande « target remote »

```

${PREFIX}gdb $BIN \
--eval-command="target remote localhost:1234" \
--eval-command="hbreak *0x$ADDR"
    
```

- Si le programme n'est pas le premier à s'exécuter au démarrage du système, point d'arrêt matériel : « hbreak » au lieu de « break ».

```

ADDR=$( ${PREFIX}nm $BIN |
sed -n "s,^\([^ ]*\).*$FUNC$, \1,p" )
    
```

- 1) tests automatisés
- 2) debug
- 3) couverture de test

- Particulièrement utile en certification DO-178
- Méthode classique d'analyse de la couverture de test
 - système instrumenté testé sur cible réelle
- Nouvelle approche
 - système réel testé sur cible instrumentée

= projet Couverture
- Qemu fournit des traces
- xcov analyse les traces
 - synthèse au niveau source ou objet

XCOV coverage report

Coverage level: branch

Trace Filename	Program	Date	Tag
explore.trace	obj/powerpc-elf/explore	2011-03-23 18:19:14	

	total nb of lines				
Total	337 lines	294 lines (87 %)	19 lines (6 %)	24 lines (7 %)	

Filename	total nb of lines				
explore.adb	19 lines	16 lines (84 %)	2 lines (11 %)	1 lines (5 %)	
b_explore.adb	25 lines	24 lines (96 %)	0 lines (0 %)	1 lines (4 %)	
actors.adb	6 lines	3 lines (50 %)	0 lines (0 %)	3 lines (50 %)	
actors.ads	3 lines	3 lines (100 %)	0 lines (0 %)	0 lines (0 %)	
queues.adb	27 lines	19 lines (70 %)	6 lines (22 %)	2 lines (7 %)	
links.adb	39 lines	33 lines (85 %)	3 lines (8 %)	3 lines (8 %)	
geomaps.adb	15 lines	15 lines (100 %)	0 lines (0 %)	0 lines (0 %)	
robots_devices.ads	5 lines	5 lines (100 %)	0 lines (0 %)	0 lines (0 %)	
robots.adb	43 lines	36 lines (84 %)	4 lines (9 %)	3 lines (7 %)	
robots.ads	2 lines	2 lines (100 %)	0 lines (0 %)	0 lines (0 %)	

```
44 + procedure Pop (Item : out Data_Type; Q : in out Queue) is
45 . begin
46 !     if Empty (Q) then
47 -         raise Program_Error;
48 .     end if;
49 .
50 +     Item := Q.Items (Q.Front);
51 !     if Q.Front = Q.Items'Last then
52 +         Q.Front := Q.Items'First;
53 .     else
54 !         Q.Front := Q.Front + 1;
55 .     end if;
56 +     Q.Size := Q.Size - 1;
57 + end Pop;
```

- Qemu : <http://wiki.qemu.org>
- Contribuer à Qemu :
http://ingenierie.openwide.fr/content/download/2472/18834/file/Qemu_article_technique.pdf
- Couverture : <http://www.projet-couverture.com>
- Fully automated bisecting :
<https://lwn.net/Articles/317154>

questions ?