# ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΥΠΟΛΟΓΙΣΤΩΝ

## Χειμερινό Εξάμηνο 2019/2020

**ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ**

**ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ**

# 1η Εργαστηριακή Άσκηση

## Εισαγωγή στον Εξομοιωτή gem5

Η πρώτη εργαστηριακή άσκηση έχει σκοπό να σας φέρει σε επαφή με ένα από τα πιο διαδεδομένα εργαλεία που χρησιμοποιούνται από την ευρύτερη κοινότητα στο χώρο της Αρχιτεκτονικής Υπολογιστών. Ο εξομοιωτής gem5 είναι ένας εξομοιωτής πλήρους συστήματος (full-system simulator), δηλαδή είναι ένα πρόγραμμα που μπορεί να εξομοιώσει τη συμπεριφορά ολοκληρωμένων υπολογιστικών συστημάτων που περιλαμβάνουν επεξεργαστές, στοιχεία μνήμης, διατάξεις αποθήκευσης και δίκτυα. Μέσω του gem5 μπορεί να περιγραφεί ένα υπολογιστικό σύστημα και στη συνέχεια να εκτελεστούν προγράμματα, ακόμα και πλήρη λειτουργικά συστήματα, πάνω σε αυτό, παρέχοντας αναλυτικές πληροφορίες για τη συμπεριφορά τους με αποτέλεσμα να καθίσταται εξαιρετικά χρήσιμος για τη μελέτη της επίδρασης σχεδιαστικών αποφάσεων. Είναι πρόγραμμα ανοικτού λογισμικού και διατίθεται ελεύθερα. Περισσότερες πληροφορίες μπορείτε να βρείτε στο site: http://gem5.org/ .

Προαπαιτούμενα

Ο gem5 είναι γραμμένος σε C/C++ και χρησιμοποιεί εκτεταμένα τη γλώσσα Python για να οριστούν τα περισσότερα configurations του. Κατά συνέπεια, απαιτείται από εσάς μια βασική εξοικείωση με τις γλώσσες αυτές.

Ο εξομοιωτής γίνεται compile και εκτελείται σε περιβάλλον Linux. Για τις ασκήσεις θα γίνει η υπόθεση ότι διαθέτετε κάποιο Ubuntu-based σύστημα (όλα που θα παρουσιαστούν έχουν δοκιμαστεί σε Ubuntu 19.10) στο οποίο έχετε root access. Παρόλα αυτά θα πρέπει να τονιστεί ότι μπορείτε να χρησιμοποιήσετε οποιαδήποτε άλλη διανομή σας βολεύει ή διαθέτετε ήδη εγκατεστημένη και στο βαθμό που είναι εφικτό θα δίνονται επαρκείς πληροφορίες ώστε να σας βοηθήσουν να διαχειριστείτε ό,τι απαιτείται και σε άλλες διανομές (π.χ. RedHat-based όπως το Fedora ή το CentOS). Κατά πάσα πιθανότητα είναι εφικτό να καταφέρετε να τρέξετε τον gem5 στα Windows μέσω Cygwin ή Windows Subsystem for Linux, όμως δεν συνίσταται και δεν θα προσφερθεί υποστήριξη για κάτι τέτοιο. Αντίθετα μέσα στους σκοπούς της άσκησης είναι και η εξοικείωση σας με τα συστήματα Linux και τα διάφορα διαθέσιμα open source εργαλεία ανάπτυξης εφαρμογών. Αν θέλετε να αποφύγετε την εγκατάσταση κάποιας διανομής Linux στον υπολογιστή σας, μπορείτε να χρησιμοποιήσετε κάποιο πρόγραμμα virtualization, όπως το VirtualBox (https://www.virtualbox.org/) ή το VMWare Workstation (μπορείτε να χρησιμοποιήσετε την δωρεάν έκδοση VMWare Workstation Player 15 την οποία έχουμε χρησιμοποιήσει και εμείς - https://my.vmware.com/en/web/vmware/free#desktop_end_user_computing/vmware_workstation_player/15_0).

Τέλος, θα χρειαστεί να μάθετε να χρησιμοποιείτε σε πολύ βασικές γραμμές το git (code versioning) και τη markup γλώσσα Markdown.

**ΜΕΡΟΣ ΠΡΩΤΟ**

---

**Βήμα 1ο.** Προετοιμασία του συστήματος και εγκατάσταση του gem5

Όπως προαναφέρθηκε, γίνεται η υπόθεση ότι διαθέτετε ένα σύστημα linux στο οποίο έχετε root access. Ο gem5 διατίθεται σε μορφή κώδικα, τον οποίο πρέπει να κατεβάσετε και να κάνετε compile. Για να μπορέσει να ολοκληρωθεί η διαδικασία του build θα πρέπει να έχετε κατεβάσει και εγκαταστήσει μια σειρά από προγράμματα και πακέτα από τα οποία εξαρτάται ο gem5 (dependencies). Για τούτο και χρειάζεστε να έχετε root access στο μηχάνημα που θα δουλεύετε – έτσι είναι εφικτή η εγκατάσταση όλων αυτών των προγραμμάτων.

Όλα που παρουσιάζονται παρακάτω έχουν ελεγχθεί σε ένα σύστημα Ubuntu 19.10 (http://releases.ubuntu.com/19.10/). Τα βασικά πακέτα από τα οποία εξαρτάται ο gem5, παρατίθενται εδώ: http://www.gem5.org/Dependencies .

Αναλυτικά θα χρειαστείτε τα παρακάτω:

- git για code versioning
- gcc / g++ έκδοση 5 και άνω
- python έκδοση 2.7 (προσοχή υπάρχουν κάποια προβλήματα ακόμα με την python 3.x)
- scons έκδοση 3.0 ή νεότερη
- zlib οποιαδήποτε σχετικά πρόσφατη έκδοση
- m4 οποιαδήποτε σχετικά πρόσφατη έκδοση
- protobuf έκδοση 2.1 και άνω
- SWIG έκδοση 2.0.4 και άνω
- Boost library
- Pydot
- libgoogle-perftools

Για την εγκατάσταση των παραπάνω, ανοίξτε ένα terminal και πληκτρολογήστε τα παρακάτω:

```
$ sudo apt install build-essential
$ sudo apt install python python-dev python-six
$ sudo apt install scons m4 swig libboost-all-dev protobuf*
$ sudo apt install libprotobuf-dev libprotoc-dev pkg-config
$ sudo apt install zlib1g zlib1g-dev zlibc
$ sudo apt install python-pip
$ pip install pydot
$ sudo apt install git cmake bison flex graphviz
$ sudo apt install libgoogle-perftools-dev
```

Χρησιμοποιώντας την εντολή sudo σημαίνει ότι η επόμενη εντολή που ακολουθεί εκτελείται με δικαιώματα διαχειριστή. Κατά συνέπεια, θα σας ζητηθεί το σχετικό password. Αν ο λογαριασμός που χρησιμοποιείτε έχει τέτοια δικαιώματα, τότε απλά εισάγετε το password σας. Αν δεν έχει, τότε θα πρέπει να συνδεθείτε με ένα λογαριασμό που έχει τέτοια δικαιώματα.

Να σημειωθεί ότι ενδέχεται ορισμένα από αυτά τα πακέτα να τα έχετε ήδη εγκατεστημένα στο σύστημα σας. Στη περίπτωση αυτή θα δείτε κάποιο σχετικό μήνυμα αλλά δεν δημιουργείται κάποιο πρόβλημα. Προσέξτε σε περίπτωση που χρησιμοποιείτε RedHat-based διανομές (Fedora, CentOS) ότι τα πακέτα ίσως έχουν διαφορετικές ονομασίες ενώ χρησιμοποιείτε το yum αντί του apt. Επίσης κάποια πακέτα μπορεί να μην υπάρχουν στα repositories και θα πρέπει να τα εγκαταστήσετε from source (συγκεκριμένα τα protobuf και swig πρέπει να γίνουν έτσι).

Αφού εγκαταστήσετε τα παραπάνω, θα ήταν καλό να κάνετε έλεγχο ποια έκδοση του gcc/g++ διαθέτετε. Πληκτρολογήστε:

```
$ gcc -v
```

Στην τελευταία γραμμή θα δείτε την έκδοση του gcc που χρησιμοποιεί by default το σύστημα σας. Όπως αναφέρθηκε θα πρέπει το σύστημα σας να έχει εγκατεστημένη έκδοση 5 και άνω. Παρόλα αυτά σε νεότερα συστήματα (όπως π.χ. το Ubuntu 19.10) η έκδοση που εγκαθίσταται by default είναι μια αρκετά καινούρια έκδοση του gcc (9.x) με την οποία είναι γνωστό ότι υπάρχουν προβλήματα κατά τη διαδικασία του compile του gem5. Κατά συνέπεια συνίσταται η εγκατάσταση μιας παλιότερης έκδοσης του gcc. Μπορείτε να χρησιμοποιήστε την έκδοση 7 με σχετική ασφάλεια. Εγκαταστήστε τη σχετική έκδοση ως εξής:

```
$ sudo apt install gcc-7 g++-7
```

Με την επιτυχή ολοκλήρωση της εγκατάστασης της έκδοσης αυτής των gcc/g++, στο σύστημα σας θα έχετε δύο εκδόσεις των gcc/g++ (εκτός και αν εσείς είχατε και άλλες). Οι εντολές που ακολουθούν θα αντιστοιχήσουν μια προτεραιότητα στη κάθε έκδοση (εδώ σαν παράδειγμα στην έκδοση 9 και στην έκδοση 7 με την 9 να έχει τη μεγαλύτερη προτεραιότητα (90)) και θα σας δώσουν έναν εύκολο τρόπο να χρησιμοποιείτε την μία ή την άλλη έκδοση ανάλογα με τις ανάγκες σας.

```
$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 90
$ sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-9 90
$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-7 70
$ sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-7 70
```

Πληκτρολογώντας τώρα:

```
$ sudo update-alternatives --config gcc
```

θα σας παρουσιαστεί η ακόλουθη έξοδος από όπου μπορείτε να επιλέξετε ποια έκδοση του gcc θέλετε να χρησιμοποιείται από το σύστημα σας:

```
There are 2 choices for the alternative gcc (providing /usr/bin/gcc).

  Selection    Path            Priority   Status
------------------------------------------------------------
  0            /usr/bin/gcc-9   90        auto mode
* 1            /usr/bin/gcc-7   70        manual mode
  2            /usr/bin/gcc-9   90        manual mode

Press <enter> to keep the current choice[*], or type selection number:
```

Ενώ, πληκτρολογώντας:

```
$ sudo update-alternatives --config g++
```

θα σας παρουσιαστεί η ακόλουθη έξοδος από όπου μπορείτε να επιλέξετε ποια έκδοση του g++ θέλετε να χρησιμοποιείται από το σύστημα σας:

```
There are 2 choices for the alternative gcc (providing /usr/bin/g++).

  Selection    Path            Priority   Status
------------------------------------------------------------
  0            /usr/bin/g++-9   90        auto mode
* 1            /usr/bin/g++-7   70        manual mode
  2            /usr/bin/g++-9   90        manual mode

Press <enter> to keep the current choice[*], or type selection number:
```

Στο σημείο αυτό θα πρέπει να κατεβάσετε και να εγκαταστήσετε τον gem5. Ο κώδικας του gem5 βρίσκεται στο εξής repository: https://gem5.googlesource.com/public/gem5 . Για να τον κατεβάσετε, θα πρέπει να κάνετε clone το repository αυτό στον υπολογιστή σας. Αυτό γίνεται ως εξής:

```
$ git clone https://gem5.googlesource.com/public/gem5 my_gem5
```

Ως `my_gem5` ορίζετε το directory στο οποίο θέλετε να γίνει clone ο κώδικας του gem5. Μπορείτε να επιλέξετε όποιο όνομα directory θέλετε, αλλά δεν μπορείτε να χρησιμοποιήσετε ένα προϋπάρχον directory. Μπορείτε να μην υποδείξετε κάποιο directory και θα δημιουργηθεί ένα καινούργιο με το όνομα `gem5`.

Όταν ολοκληρωθεί η διαδικασία, θα έχει δημιουργηθεί ένα αντίγραφο του κώδικα του gem5 (καθώς και όλων των υποστηρικτικών ή άλλων αρχείων του repository) στο φάκελο που έχετε ορίσει. Για να κάνετε build τον gem5 θα χρησιμοποιήσετε το SCons.

Μπορείτε να κάνετε build τον gem5 ώστε να υποστηρίζει επεξεργαστές διαφορετικών συνόλων εντολών. Στη συγκεκριμένη εργασία θα επικεντρωθείτε στην έκδοση του gem5 για ARM ISAs (Instruction Set Architectures). Επιπλέον υπάρχει η δυνατότητα να κάνετε build τον gem5 με διαφορετικά χαρακτηριστικά που επιτρέπουν την πιο εύκολη αποσφαλμάτωση του ή την πιο γρήγορη εκτέλεση της εξομοίωσης. Στην εργασία αυτή θα κάνετε build την έκδοση που διατηρεί μια ισορροπία ανάμεσα στις πληροφορίες που παρέχει και στην ταχύτητα εκτέλεσης, την gem5.opt. Αφού μπείτε στο directory που είναι τα αρχεία του gem5, πληκτρολογήστε την ακόλουθη εντολή αλλάζοντας το N με τον αριθμό των επεξεργαστών που έχει ο υπολογιστής σας ώστε να χρησιμοποιήσετε τη μέγιστη παραλληλία που μπορείτε και να επιταχύνετε τη διαδικασία.

```
$ scons build/ARM/gem5.opt -j N
```

Η διαδικασία του build απαιτεί κάποια ώρα για να ολοκληρωθεί. Όταν τελειώσει θα έχετε έτοιμο τον gem5 ώστε να μπορέσετε να εκτελέσετε τα πρώτα σας προγράμματα.

**Βήμα 2ο**. Γνωριμία με τον gem5 και εκτέλεση ενός “Hello World” παραδείγματος.

Η γενική μορφή της εντολής που χρησιμοποιείτε για να εκτελέσετε τον gem5 είναι η ακόλουθη (με <...> είναι τα απαραίτητα στοιχεία και με […] τα προαιρετικά):

```
$ <gem5_binary> [gem5_options] <simulation_script> [script_options]
```

Χρησιμοποιώντας το flag -h μπορείτε να δείτε τις διαθέσιμες επιλογές για τον gem5 και για το simulation script που χρησιμοποιείτε.

```
$ <gem5_binary> -h (για τις διαθέσιμες επιλογές για τον gem5)

$ <gem5_binary> [gem5_options] <simulation_script> -h (για τις διαθέσιμες επιλογές του simulation script)
```

Το simulation script που αναφέρεται παραπάνω είναι ένα python script το οποίο ορίζει τις βασικές παραμέτρους της εξομοίωσης και εκτελεί τη διαδικασία. Έτσι μέσω αυτού μπορείτε να ορίσετε παραμέτρους όπως CPUs, caches, memory controllers κτλ κτλ. Μια σειρά από βασικά scripts τα οποία μπορείτε να χρησιμοποιήσετε υπάρχουν ήδη μέσα στα αρχεία που έχετε κατεβάσει και βρίσκονται στον κατάλογο configs/example/ .

Ο gem5 υποστηρίζει δύο βασικά simulation modes. Το πρώτο (και αυτό που θα χρησιμοποιήσετε στην εργασία αυτή) ονομάζεται System call Emulation (SE) στο οποίο ο εξομοιωτής επικεντρώνεται στον επεξεργαστή και το υποσύστημα μνήμης και όχι στο πλήρες σύστημα. Κατά συνέπεια, σε αυτό το mode ο gem5 δεν τρέχει ένα πλήρες λειτουργικό σύστημα, αλλά μια εφαρμογή που του ορίζετε εσείς και κάνει emulate όλα τα system calls που ενδεχομένως ανακύπτουν.

Στο Full System (FS) mode, ο gem5 πραγματοποιεί εξομοίωση ενός πλήρους συστήματος και εκτελεί ένα πλήρες λειτουργικό σύστημα. Ο χρήστης μπορεί να συνδεθεί κανονικά στο σύστημα αυτό και να το χρησιμοποιήσει με τον ίδιο ακριβώς τρόπο που χρησιμοποιεί και το λειτουργικό σύστημα σε ένα κανονικό υπολογιστή και φυσικά μπορεί να τρέξει εφαρμογές μέσα στο περιβάλλον αυτό. Ο gem5 γενικά υποστηρίζει λειτουργικά συστήματα Linux-based (για παράδειγμα μπορεί να εκτελέσει και Android που βασίζεται στον Linux kernel).

Εκτελέστε την ακόλουθη εντολή:

```
$ ./build/ARM/gem5.opt configs/example/arm/starter_se.py --cpu="minor"
"tests/test-progs/hello/bin/arm/linux/hello" # Hello world!
```

Εκτελώντας την εντολή αυτή, ο gem5 θα τρέξει ένα precompiled πρόγραμμα (αυτό που βρίσκεται στα εισαγωγικά), με τις παραμέτρους που ορίζονται στο starter_se.py script για το μοντέλο minor μοντέλο CPU. Τα σημαντικότερα που χρειάζεται να κρατήσετε από την έξοδο που παράγεται είναι τα ακόλουθα:

```
Global frequency set at 1000000000000 ticks per second

info: Entering event queue @ 0.  Starting simulation...

Hello world!

exiting with last active thread context  @  24087000
```

Η πρώτη γραμμή δίνει πληροφορίες για τη συχνότητα λειτουργίας που έχει οριστεί για τον επεξεργαστή. Ο gem5 αντιστοιχίζει ticks σε picoseconds, κατά συνέπεια η συχνότητα που αναφέρεται αντιστοιχεί σε 1GHz.

Το "`Hello world!`" είναι η έξοδος του προγράμματος που μόλις εκτελέσατε!

Η τελευταία γραμμή μας πληροφορεί πόσα ticks χρειάστηκαν για να ολοκληρωθεί η εκτέλεση του προγράμματος.

Τα ενδιαφέροντα αποτελέσματα για τη χρήση του gem5 ως εργαλείο για την αρχιτεκτονική υπολογιστών παρόλα αυτά δεν βρίσκονται στην έξοδο που μόλις παρατήσατε. Αντίθετα, τα στοιχεία που βασικά ενδιαφέρουν είναι τα στατιστικά που προκύπτουν από την εκτέλεση του προγράμματος. Εφόσον δεν έχετε ορίσει κάποιο άλλο φάκελο κατά την κλήση του gem5, τα αποτελέσματα αυτά αποθηκεύονται στο φάκελο m5out. Εκεί θα πρέπει να βρείτε μεταξύ άλλων το αρχείο `stats.txt` που περιέχει όλα τα στατιστικά στοιχεία της εκτέλεσης.

Προσοχή! Αν δεν ορίσετε κάποιο φάκελο στον οποίο θα γραφτούν τα αποτελέσματα, τότε όπως αναφέρεται παραπάνω, τα αποτελέσματα θα γράφονται στον ίδιο default φάκελο (m5out), οπότε κάθε εκτέλεση θα έχει αποτέλεσμα να αντικαθιστά τα αποτελέσματα της προηγούμενης. Χρησιμοποιήστε την επιλογή -d <directory_name> κατά τη κλήση του gem5 ώστε να ορίσετε που θα γραφτούν τα αποτελέσματα. Παράδειγμα εκτέλεσης με τα αποτελέσματα να γράφονται στον φάκελο hello_result:

```
$ ./build/ARM/gem5.opt -d hello_result configs/example/arm/starter_se.py
--cpu="minor" "tests/test-progs/hello/bin/arm/linux/hello"
```

**Ερωτήματα Πρώτου Μέρους**

1. Ανοίξτε το αρχείο starter_se.py που χρησιμοποιήσατε στο παράδειγμα του Hello World και προσπαθήστε να καταλάβετε ποιες είναι βασικές παράμετροι που έχει περάσει στον gem5 για το σύστημα προς εξομοίωση. Καταγράψτε τα βασικά χαρακτηριστικά του συστήματος, όπως τύπος CPU, συχνότητα λειτουργίας, βασικές μονάδες, caches, μνήμη κτλ.
2. Εκτός από το αρχείο εξόδου stats.txt που παράγεται στο τέλος της εξομοίωσης, ο gem5 παράγει και τα αρχεία config.ini και config.json. Τα αρχεία αυτά παρέχουν πληροφορίες για το σύστημα που εξομοιώνει ο gem5.
    a. Χρησιμοποιήστε τα αρχεία αυτά για να επαληθεύσετε την απάντηση σας στο πρώτο ερώτημα. Παραθέστε τα σχετικά πεδία.
    b. Ποιό είναι το συνολικό νούμερο των «committed» εντολών? Γιατί δεν είναι ίδιο το νούμερο με αυτό που παρουσιάζεται από στατιστικά που παρουσιάζονται από τον gem5?
    c. Πόσες φορές προσπελάστηκε η L2 cache? Πώς θα μπορούσατε να υπολογίσετε τις προσπελάσεις αν δεν παρεχόταν από τον εξομοιωτή?
3. Εκτός από τις πληροφορίες που παρέχονται σε αυτήν την άσκηση, είναι σημαντικό να μπορείτε να ανατρέχετε και να αναζητάτε πληροφορίες στη βιβλιογραφία. Έτσι χρησιμοποιώντας ως αρχή το site του gem5 (gem5.org) αναζητήστε πληροφορίες για τα διαφορετικά μοντέλα in-order CPUs που χρησιμοποιεί ο gem5 (hint: στο παράδειγμα χρησιμοποιήσατε το μοντέλο CPU: minor) και παραθέστε μια συνοπτική παράγραφο για καθένα από αυτά.
    a. Γράψτε ένα πρόγραμμα σε C και εκτελέστε το στον gem5 χρησιμοποιώντας διαφορετικά μοντέλα CPU και κρατώντας όλες τις άλλες παραμέτρους ίδιες. Χρησιμοποιήστε τα TimingSimpleCPU και MinorCPU.

(Παρατήρηση: Μην χρησιμοποιήσετε τα ίδιο configuration αρχείο (starter_se.py) αλλά το αρχείο configs/example/se.py . Παράδειγμα εκτέλεσης:

```
$ ./build/ARM/gem5.opt configs/example/se.py --cpu=MinorCPU –caches
tests/test-progs/hello/bin/arm/linux/hello
```

Παραθέστε τα αποτελέσματα σας όσον αφορά τους χρόνους εκτέλεσης (hint: stats.txt). Προσοχή, ο gem5 είναι πολύ αργότερος στην εκτέλεση ενός προγράμματος από ότι ο επεξεργαστής σας όταν το τρέχετε κατευθείαν σε αυτόν. Συνεπώς μην φτιάξετε κάποιο εξαιρετικά απαιτητικό πρόγραμμα. [*]

b. Αν τα αποτελέσματα που παρατηρείτε διαφέρουν, με βάση όσα περιγράψατε για τα χαρακτηριστικά κάθε μοντέλου, δώστε μια εξήγηση των διαφορών που παρατηρείτε. Αντίστοιχα, για τα όμοια αποτελέσματα δικαιολογήστε γιατί τα σχετικά μοντέλα παράγουν το ίδιο αποτέλεσμα.
c. Αλλάξτε μια παράμετρο του επεξεργαστή και παρατηρήστε τα αποτελέσματα για τα δύο διαφορετικά CPU models. Δοκιμάστε να αλλάξετε την συχνότητα λειτουργίας και τη τεχνολογία της μνήμης που χρησιμοποιείτε. Παραθέστε και δικαιολογήσετε τα αποτελέσματα που παρατηρήσατε.

[*] Πώς να κάνετε compile ένα πρόγραμμα για τον gem5 που υποστηρίζει ARM αρχιτεκτονική στον υπολογιστή σας.

Η διανομή Linux που χρησιμοποιείτε τρέχει σε επεξεργαστή x86, συνεπώς και οι compilers που είναι εγκατεστημένοι παράγουν binaries τα οποία είναι φτιαγμένα για την αρχιτεκτονική αυτή. Ο gem5 όμως που χρησιμοποιείτε σε αυτήν την εργασία εξομοιώνει έναν ARM επεξεργαστή και άρα δεν μπορεί να τρέξει binaries άλλης αρχιτεκτονικής.

Αυτό είναι ένα κοινό πρόβλημα όταν αναπτύσσετε λογισμικό για μια πλατφόρμα που δεν είναι ίδια με αυτήν στην οποία το αναπτύσσετε. Για παράδειγμα, αν χρησιμοποιείτε τον υπολογιστή σας (που έχει κάποιο επεξεργαστή της Intel ή της AMD) για να γράψετε κάποιο λογισμικό για το Raspberry ή το κινητό σας (που έχει κάποιο ARM επεξεργαστή). Η λύση είναι η χρήση cross compilers, δλδ compilers που παράγουν binaries για κάποια άλλη πλατφόρμα. Στη συγκεκριμένη περίπτωση θα χρειαστείτε τους cross compilers για ARM επεξεργαστές. Σε Ubuntu διανομές τα σχετικά πακέτα είναι άμεσα διαθέσιμα στα repositories και απλά χρειάζεται να κάνετε τα ακόλουθα για να εγκαταστήσετε το ARM cross compiler toolchain[1]:

```
$ sudo apt install gcc-arm-linux-gnueabihf
$ sudo apt install g++-arm-linux-gnueabihf
```

Επειδή ο gem5 στην έκδοση SE δεν τρέχει κάποιο λειτουργικό, δεν μπορεί να κάνει linking με dynamic libraries. Αυτό σημαίνει ότι πρέπει να κάνετε στατικά compile το πρόγραμμα σας, χρησιμοποιώντας κατά το compilation το flag `--static`. Υποθέτοντας ότι ο κώδικας σας είναι στο αρχείο myprog.c, θα πρέπει να δώσετε μια τέτοιου τύπου εντολή:

[1] Σε άλλες διανομές ίσως να χρειάζεται να αναζητήσετε ξεχωριστά τα πακέτα αυτά και να τα κατεβάσετε σε μορφή source ή κάποιου εγκαταστάσημου πακέτου (π.χ. .rpm).

```
$ arm-linux-gnueabihf-gcc --static myprog.c -o myprog_arm
```

Αφού ολοκληρώσετε επιτυχώς τη διαδικασία, θα πρέπει να τρέξετε τον gem5 δίνοντας του ως argument το πρόγραμμα που παράξατε. Η διαδικασία είναι ίδια με το πρόγραμμα hello που τρέξατε στο παράδειγμα.

**Παραδοτέα**

Ανοίξτε (αν δεν διαθέτετε ήδη) έναν λογαριασμό στο GitHub. Δημιουργήστε ένα public repository και ανεβάστε:

1. ό,τι κώδικες και αρχεία αποτελεσμάτων έχουν προκύψει από τα ερωτήματα που σας ζητούνται να απαντήσετε. Για τους κώδικες που θα ανεβάσετε απαιτείται να έχετε οπωσδήποτε επαρκή σχόλια που να επεξηγούν τι έχετε κάνει και αν αυτό απαιτείται κάποιο documentation.
2. μια αναλυτική αναφορά με τις απαντήσεις στα ερωτήματα. Η αναφορά θα είναι γραμμένη σε ένα αρχείο README.md που θα είναι στο top level του repository σας. Η αναφορά σας θα πρέπει να περιλαμβάνει οπωσδήποτε τις πηγές που χρησιμοποιήσατε για να ολοκληρώσετε την εργασία σας.

Για να παραδώσετε την άσκηση σας θα πρέπει να χρησιμοποιήσετε όπως καταλαβαίνετε το git και τη γλώσσα Markdown για την αναφορά σας. Και τα δύο είναι εξαιρετικά απλά και εξαιρετικά χρήσιμα. Ένα πολύ καλό άρθρο για το git μπορείτε να βρείτε εδώ: https://www.freecodecamp.org/news/the-essential-git-handbook-a1cf77ed11b5/ και ένα εξαιρετικά απλό ξεκίνημα για τη γλώσσα Markdown εδώ: https://www.markdowntutorial.com/ .

ΣΗΜΑΝΤΙΚΟ: Όταν ολοκληρώσετε επιτυχώς την εργασία σας, μπορείτε να πάρετε έναν έξτρα βαθμό, γράφοντας μια κριτική για την εργασία αυτή. Επικεντρωθείτε στο αν μάθατε κάτι, αν τη θεωρείτε απλοϊκή ή δύσκολη, αν υπήρχε κάτι που σας δυσκόλεψε χωρίς λόγο και χωρίς να είναι σχετικό με την εργασία σας, κάποια συμβουλή κτλ. Βάλτε αυτή την κριτική μέσα στην αναφορά σας που είναι στο README.md

Για την παράδοση της άσκησης σας, θα κάνετε submit μέσω του elearning την διεύθυνση του repository σας (όχι τον κώδικα ή οτιδήποτε άλλο ΜΟΝΟ την διεύθυνση στο github).

Να έχετε υπόψη σας τα εξής:

1. εφόσον όλα τα repositories είναι public, όλοι μπορούν να δουν το κώδικα σας και την εργασία σας. Αυτό σημαίνει ότι θα πρέπει να φροντίσετε να είναι προσεγμένα (θα κριθείτε και για την ποιότητα της εργασίας όσον αφορά την παρουσίαση της)
2. με την ίδια λογική εφόσον όλα είναι public γίνεται προφανές ότι είναι εξαιρετικά εύκολο να αναδειχθούν οι αντιγραφές. Καλό θα ήταν λοιπόν να το λάβετε αυτό υπόψη σας.
3. Η πρώτη αυτή άσκηση είναι εξαιρετικά απλή αλλά έχει στόχο να σας εξοικειώσει με πολλά διαφορετικά πράγματα, όπως η επαφή σας με το Linux, η χρήση διαδεδομένων εργαλείων για την ανάπτυξη κώδικα, cross compilation, εργαλεία αρχιτεκτονικής, αναζήτηση πληροφοριών. Εκμεταλλευτείτε το για να πειραματιστείτε και να μάθετε όσο μπορείτε καλύτερα τα εργαλεία αυτά.

Για απορίες σε σχέση με τον GEM5 και μόνο (ΟΧΙ με την VM ή οτιδήποτε άλλο) μπορείτε να στέλνετε email στο dkaranassos@ece.auth.gr.

ΚΑΛΗ ΕΠΙΤΥΧΙΑ