ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

Ενσωματωμένα Συστήματα

Ενότητα: ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ ANDROID-DEVKIT No:03

Δρ. Μηνάς Δασυγένης

mdasyg@ieee.org

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών

http:// arch.icte.uowm.gr/mdasyg

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα του Πανεπιστημίου Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Περιεχόμενα

1. Σκοπός της άσκησης

- Μεταγλώττιση του android για την αναπτυξιακή πλακέτα beagleboard.
- Τροποποίηση του u-boot loader.
- Αποσφαλμάτωση με το adb, μεταφορά αρχείων και προβολή συμβάντων.

2. Παραδοτέα

(A) 2 ερωτήσεις
(C) 5 ασκήσεις

- **Παραδοτέο C1:** Screenshot στο οποίο να φαίνεται το επιτυχές χτίσιμο του λειτουργικού για την πλακέτα Beagleboard.
- **Παραδοτέο C2:** Screenshot στο οποίο να φαίνεται η επιτυχής δημιουργία των απαιτούμενων διαμερισμάτων στη κάρτα SD.
- **Παραδοτέο C3:** Screenshot στο οποίο να φαίνεται η σειριακή σύνδεση που έχει επιτευχθεί μεταξύ της πλακέτας και του λειτουργικού συστήματος.
- **Παραδοτέο C4:** Screenshot στο οποίο να φαίνεται η αλλαγή των μεταβλητών περιβάλλοντος προκειμένου οι παραπάνω αλλαγές να γίνουν μόνιμες.
- **Παραδοτέο C5:** Screenshot στο οποίο να φαίνονται οι καταγραφές απο το MyApp.

Σημείωση:
Τα εργαστήρια απαιτούν την ύπαρξη λειτουργικού συστήματος Linux. Θα πρέπει να γίνει κανονική εγκατάσταση του λειτουργικού και όχι εκτέλεση μέσω εικονικής μηχανής καθώς κάτι τέτοιο δεν υποστηρίζεται. Η παρούσα εργασία έχει υλοποιηθεί σε Ubuntu 10.4 64bit, το οποίο και προτείνεται.

Το υλικό του εργαστηρίου αναπτύχθηκε σε συνεργασία με τον κ. Τάσο Φετινίδη.

Χρησιμοποιήθηκε υλικό από http:// free-electrons.com .

3. Περιγραφή της εργαστηριακής άσκησης

3.1 Χτίστε το Android για την πλακέτα BeagleBoard

Επιστρέψτε στον κατάλογο **src**.

Όπως αναφέραμε πιο πριν, η πλακέτα **DevKit8000** είναι παρόμοια με την BeagleBoard. Έτσι, αρχικά θα ασχοληθούμε με το compile του BeagleBoard. Η ζητούμενη εντολή είναι:

```
make TARGET_PRODUCT=beagleboard TARGET_TOOLS_PREFIX=
~/felabs/android/linaro/android-toolchain-eabi/bin/arm-
eabi-
boottarball systemtarball userdatatarball -j8
```

Παραδοτέο C1: Screenshot στο οποίο να φαίνεται το επιτυχές χτίσιμο του λειτουργικού για την πλακέτα Beagleboard.

Παρόμοια, μη ξαφνιαστείτε αν η διαδικασία πάρει πάρει πάνω από ώρα μέχρι να ολοκληρωθεί, ακόμα και σε ένα πρόσφατης τεχνολογίας φορητό υπολογιστή.

Η διαδικασία αυτή θα δημιουργήσει τρια tarballs στον κατάλογο **inout/target/product/beagleboard** με ονόματα **boot.tar.bz2**, **system.tar.bz2** και **userdata.tar.bz2**. Ακόμα, θα δημιουργηθούν οι εικόνες του Xloader, του U-Boot και του πυρήνα Linux.

Ερώτηση Α1: Ποια είναι η λειτουργία του κάθε tarball; Πως χρησιμοποιούνται αυτά από το λειτουργικό σύστημα;

Πρέπει να τοποθετήσουμε τις τρεις αυτές εικόνες στην κάρτα SD, έτσι ώστε να μπορούμε να φορτώσουμε το λειτουργικό σύστημα στην πλακέτα

Αρχικά, πάρτε την SD κάρτα που σας παρείχε ο καθηγητής και τοποθετήστε την στη κατάλληλη θύρα ανάγνωσης τέτοιων καρτών ή αντίστοιχα σε έναν usb sd αναγνώστη μνημών που ενδεχομένως σας έδωσε ο καθηγητής.

Μετά, χρησιμοποιώντας την εντολή **dmesg** βρείτε ποια συσκευή στον σταθμό εργασίας χρησιμοποιεί την κάρτα SD. Ας υποθέσουμε ότι είναι η **/dev/sdc.**

Τώρα, μπορούμε να χρησιμοποιήσουμε το σκριπτάκι alinaro-android-media-create που έχει αναπτυχθεί από το Linaro και παίρνει τα τρία tarballs και τα συνενώνει προκειμένου να δημιουργήσει ένα σύστημα έτοιμο προς φόρτωση στην κάρτα SD.

Μπορείτε να εγκαταστήσετε το παραπάνω σκριπτάκι ακολούθως:

```
bzr branch lp:linaro-image-tools

bzr revert -r2011.11

sudo ./linaro-image-tools/linaro-android-media-create --mmc
/dev/sdc

--dev beagle --system
out/target/product/beagleboard/system.tar.bz2

--userdata out/target/product/beagleboard/userdata.tar.bz2

--boot out/target/product/beagleboard/boot.tar.bz2
```

Ερώτηση Α2: Στις παραπάνω εντολές, χρησιμοποιήθηκε το πρόγραμμα bzr. Τι κάνει αυτό το πρόγραμμα; Μπορεί να αντικατασταθεί με κάποιο άλλο;

Όταν ολοκληρωθούν οι παραπάνω εντολές, μπορείτε να απομακρύνετε την κάρτα SD και να την τοποθετήσετε στην αντίστοιχη εσοχή της πλακέτας DevKit 8000.

Παραδοτέο C2: Screenshot στο οποίο να φαίνεται η επιτυχής δημιουργία των απαιτούμενων διαμερισμάτων στη κάρτα SD.

3.2 Εγκαθιστώντας την σειριακή επικοινωνία με την πλακέτα

Για να δούμε την πλακέτα να φορτώνει το λειτουργικό θα πρέπει να διαβάσουμε τα αρχικά μηνύματα που εμφανίζονται μέσω της σειριακής θύρας της πλακέτας. Ο καθηγητής θα σας παρέχει με το απαραίτητο σειριακό καλώδιο για την πλακέτα μαζί με έναν μετατροπέα usb σε σειριακό για την σύνδεση με τον φορητό υπολογιστή σας.

Όταν θα τοποθετήσετε αυτόν τον μετατροπέα, μια σειριακή θύρα θα πρέπει να εμφανιστεί στον σταθμό εργασίας σας: **/dev/ttyUSB0**

Μπορείτε επίσης να δείτε την συσκευή κοιτώντας τα αποτελέσματα της εντολής dmesg. Για να επικοινωνήσετε με την πλακέτα μέσω της σειριακής θύρας θα πρέπει να εγκαταστήσετε ένα πρόγραμμα σειριακής επικοινωνίας, όπως το picocom:

```
sudo apt-get install picocom
```

Στη συνέχεια, εκκινήστε το πρόγραμμα με τα ακόλουθα ορίσματα για να εγκαθιδρυθεί η σειριακή σύνδεση:

```
picocom -b 115200 /dev/ttyUSB0
```

στη συσκευή **/dev/ttyUSB0** και με ρυθμό μετάδοσης 115200.

Αν θέλετε να διακόψετε το πρόγραμμα δώστε στο τερματικό **[Ctrl][a]** και μετά **[Ctrl][x].**

3.3 Εκκινώντας την πλακέτα

Όταν τοποθετήσετε κατάλληλα την SD κάρτα, μπορείτε να εκκινήσετε την πλακέτα πιέζοντας το πλήκτρο boot ενώ την ανοίγετε.

Στη σειριακή θύρα, μπορείτε να δείτε το λειτουργικό Android να φορτώνει μέχρι τελικά να έχετε ένα κέλυφος στη σειριακή σύνδεση.

Ωστόσο, όπως θα έχετε παρατηρήσει, το σύστημα φορτώνει αλλά μπορεί να μην έχετε καμία ένδειξη. Θα το διορθώσουμε.

Παραδοτέο C3: Screenshot στο οποίο να φαίνεται η σειριακή σύνδεση που έχει επιτευχθεί μεταξύ της πλακέτας και του λειτουργικού συστήματος.

3.4 Διορθώνοντας την μαύρη οθόνη

Το πρώτο πρόβλημα που συναντάμε είναι ότι η οθόνη παραμένει μαύρη καθ' όλη την διαδικασία. Αυτό οφείλεται στο δημιουργημένο U-Boot που αφορά την πλακέτα BeagleBoard και όχι την DevKit8000.

Στο σύστημα δημιουργίας Android, όλες οι σχετικές ρυθμίσεις με το hardware βρίσκονται στο αρχείο BoardConfig.mk.

Στο φυλλάδιο οδηγιών της πλακέτας BeagleBoard, βρείτε την τοποθεσία του αρχείο ρυθμίσεων U-boot που χρησιμοποιείται και αλλάξτε την επιλογή, έτσι ώστε να χρησιμοποιεί τις προκαθορισμένες ρυθμίσεις για την πλακέτα DevKit8000 *(devkit8000_config)*.

Μεταγλωττίστε και δοκιμάστε με τις νέες αλλαγές. Θα πρέπει να δείτε την οθόνη να δουλεύει, ενώ έχει μια σημαντική δυσλειτουργία: τυπώνει μόνο ένα τμήμα από την οθόνη.

Μπορείτε να αποφύγετε να κάνετε μια πλήρη ανοικοδόμηση απλά αφαιρώντας τον κατάλογο out/target/product/beagleboard/obj/u-boot .

3.5 Διορθώστε την ανάλυση

Η πραγματική ανάλυση της οθόνης είναι 640x480, ενώ η ανάλυση είναι μόλις 480x272.

Αυτού του είδους οι ρυθμίσεις γίνονται κυρίως μέσω της γραμμής εντολών του πυρήνα. Στο διαμέρισμα της κάρτας SD, θα βρείτε ένα αρχείο με όνομα namedboot.txt, το οποίο είναι ένα σκριπτάκι τύπου U-Boot που θέτει όλες τις παραμέτρους που χρειάζονται για να φορτώσει η πλακέτα σωστά.

Προσθέστε το omapdss.def_disp και αλλάξτε τις ρυθμίσεις στο omapfb.mode, έτσι ώστε να χρησιμοποιεί την LCD και όχι την DVI έξοδο και τη σωστή ανάλυση.

Μπορείτε να βρείτε κάποια τεκμηρίωση για αυτές τις επιλογές στον κατάλογο kernel/Documentation/arm/OMAP/DSS στο τμήμα με τις ρυθμίσεις φόρτωσης του πυρήνα.

Θα πρέπει, ύστερα, να δημιουργήσετε από το αρχείο boot.txt file ένα αρχείο με όνομα boot.scr με την παρακάτω εντολή:

```
mkimage -A arm -O linux -T script -C none -a 0 -e 0
-n 'Execute uImage.bin' -d boot.txt boot.scr
```

Όταν τελειώσετε με αυτό το βήμα, δοκιμάστε τις αλλαγές εκκινώντας την πλακέτα. Τώρα, θα πρέπει να δείτε την οθόνη να λειτουργεί κανονικά.

3.6 Διορθώνοντας την οθόνη αφής

Αν τυχόν δοκιμάσατε την οθόνη αφής με την προηγούμενη Android κατασκευή που κάνατε, θα προσέξατε ότι η οθόνη είναι σχεδόν άχρηστη, ενώ το σύστημα λαμβάνει κάποια δεδομένα.

Αυτό σημαίνει ότι ο πυρήνας έχει τον οδηγό για τον ελεγκτή της οθόνης αφής αλλά επιστρέφει περίεργες τιμές.

Ωστόσο, αν προσέξατε, θα είδατε ότι η οθόνη αφής δεν έχει την ίδια διάταξη με την οθόνη.

Επιστρέψτε στη βοήθεια τεκμηρίωσης OMAPDSS για να βρείτε κάποια επιλογή που πιθανώς να επιλύσει το πρόβλημα σας.

Όλα σε ένα

Αυτές οι αλλαγές δεν είναι μόνιμες επειδή το αρχείο boot.scr γράφεται από την αρχή κάθε φορά από το σκριπτάκι **linaro-android-media-create**.

Ωστόσο, το σύστημα δημιουργίας μπορεί να μας βοηθήσει εδώ.

Χρησιμοποιήστε την μεταβλητή **BOARD_KERNEL_CMDLINE** για να θέσετε αυτές τις νέες τιμές.

Θα επεξεργαστεί από το σύστημα οικοδόμησης και μετά χρησιμοποιείται όταν δημιουργείται η εικόνας που φορτώνουμε.

Από την στιγμή που η μεταβλητή σχετίζεται με το hardware θα πρέπει να βρίσκεται στο αρχείο BoardConfig.mk

Μπορείτε επίσης να προσθέσετε την σημαία no_console_suspend στα ορίσματα που δίνονται στη φόρτωση καθώς το λειτουργικό Android εξ ορισμού αναστέλλει τον φλοιό στη σειριακή γραμμή μετά από περίπου ένα λεπτό και έτσι το καθιστά δύσκολο στη σωστή χρήση.

Παραδοτέο C4: Screenshot στο οποίο να φαίνεται η αλλαγή των μεταβλητών περιβάλλοντος προκειμένου οι παραπάνω αλλαγές να γίνουν μόνιμες.

3.7 Προσθέστε τα κουμπιά

Αν έτυχε να τρέξετε κάποια εφαρμογή θα παρατηρήσατε ότι δεν μπορείτε να επιστρέψετε στην αρχική οθόνη καθώς κανένα κουμπί δεν αντιστοιχηθεί για τα πλήκτρα BACK και HOME.

Η αντιστοιχία των κουμπιών γίνεται σε δύο βήματα. Αρχικά, στον πυρήνα, η πλακέτα θα πρέπει να ορίσει τα διαθέσιμα κουμπιά.

Σε αυτή την περίπτωση, θα χρησιμοποιήσουμε τα μικρά μαύρα κουμπιά που βρίσκονται ακριβώς δίπλα από την οθόνη στη πλακέτα **DevKit8000**.

Αυτά τα κουμπάκια χειρίζονται από τον οδηγό gpio-keys και ορίζονται στο αρχείο της πλακέτας devkit8000 στον πυρήνα και συγκεκριμένα στον κατάλογο **arch/arm/mach-omap2 .**

Αν κοιτάξετε σε αυτό το αρχείο, θα δείτε ότι μόνο ένα κουμπί ορίζεται, αυτό που αντιστοιχεί στην ετικέτα USER πάνω στην πλακέτα.

Θα χαρτογραφήσουμε αυτό το κουμπί ως το πλήκτρο BACK στο Android.

Στη δομή gpio_keys_button, υπάρχει ένα πεδίο κωδικού.

Αυτή η τιμή του πεδίου ορίζει το keycode που στέλνεται στο υποσύστημα εισόδου όταν πατάτε το κουμπί, και αργότερα αποστέλλεται στο userspace.

Αντικαταστήστε το keycode που χρησιμοποιείται από το KEY_EXIT και αναζητήστε την τιμή του στο αρχείο include/linux/input.h.

Το σύστημα εισόδου του λειτουργικού Android φορτώνει τα keymaps και τα key layouts για κάθε οδηγό – driver εισόδου που φορτώνεται.

Για να τα φορτώσει σωστά, χρησιμοποιεί το ίδιο όνομα με τον driver εισόδου, με μια προέκταση.

Στη περίπτωση μας, ο driver εισόδου είναι ο gpio-keys και θα πρέπει να τροποποιήσουμε το αρχείο gpio-keys.kl

Αυτό το αρχείο αποτελείται από μια λίστα με εγγραφές που αντιστοιχούν στην εκάστοτε εντολή που πρέπει το κάθε keycode να πυροδοτήσει στο σύστημα.

Προσθέστε μια καινούργια εγγραφή για το κουμπί BACK κάπως έτσι:

<keycode> BACK

Μόλις τελειώσετε, ξανά χτίστε το σύστημα, φορτώστε το και θα πρέπει πλέον να είστε σε θέση να χρησιμοποιήσετε το κουμπί BACK.

3.8 Χρησιμοποιήστε το ADB

Μετά από αυτό το εργαστήριο, θα είστε ικανοί να:

- ✓ Αποσφαλματώσετε το σύστημα και τις εφαρμογές.
- ✓ Έχετε ένα κέλυφος σε κάθε συσκευή.
- ✓ Ανταλλάσσετε αρχεία με μια συσκευή.
- ✓ Εγκαθιστάτε νέες εφαρμογές.

3.9 Εγκατάσταση

Παραμείνετε στον κατάλογο **/home/<user>/felabs/android/linaro**

3.10 Λήψη ADB

Το ADB συνήθως διανέμεται από την Google ως μέρος του δικού τους SDK για το Android.

Αν θέλετε να το χρησιμοποιήσετε θα πρέπει πρώτα να μεταβείτε στη σελίδα http://developer.android.com/sdk/ και να λάβετε το SDK για Linux και να το αποσυμπιέσετε.

Ύστερα, θα τρέχατε το σκριπτάκι tools/android και στο παράθυρο των Packages, θα επιλέγατε Android SDK Platform-tools κάτω από το Tools.

Απο-επιλέξτε όλα τα άλλα πακέτα και κάντε κλικ στο κουμπί Install 1 package. Μόλις τελειώσει, θα έχετε εγκατεστημένο το adb στον κατάλογο: **./platform-tools/adb.**

Ωστόσο, ο πηγαίος κώδικας του Android έχει και αυτός ένα ενσωματωμένο SDK, και έτσι έχετε ήδη το adb στον κατάλογο **out/host/linux-x86/bin/.**

Για να προσθέσετε αυτόν τον κατάλογο στο PATH, μετά την εγκατάσταση του Linaro που έγινε στο προηγούμενο εργαστήριο, μπορείτε να το κάνετε μέσα από το ίδιο περιβάλλον εργασίας όπως νωρίτερα:

```
source build/envsetup.sh
lunch
```

... και επιλέξτε beagleboard-eng στο lunch μενού.

3.11 Απενεργοποιήστε την αναστολή USB

Ο πυρήνας που έχουμε χτίσει χρησιμοποιεί την αναστολή των θυρών USB για εξοικονόμηση ενέργειας. Αυτή θα είναι μια επώδυνη διαδικασία κατά την διάρκεια αυτού και των επόμενων εργαστηρίων.

Μεταβείτε στον κατάλογο του πυρήνα. Αρχικά, χρησιμοποιήστε το αρχείο android_omap3_defconfig για τις αρχικές ρυθμίσεις.

Φορτώστε το με την εντολή:

```
ARCH=arm make android_omap3_defconfig
```

και μετά τροποποιήστε το με την εντολή:

```
ARCH=arm make menuconfig
```

για να απενεργοποιήσετε το CONFIG_USB_SUSPEND και να ενεργοποιήσετε το CONFIG_USB_OTG_WAKELOCK.

Θα πρέπει επίσης να ενεργοποιήσουμε την υποστήριξη **OHCI8**

(USB_OHCI_HCD_OMAP3).

Μόλις τελειώσετε, κλείστε το αρχείο και το εργαλείο ρυθμίσεων θα σώσει το νέο αρχείο ως .config.

Μπορείτε να το σώσετε σε ένα αρχείο defconfig χρησιμοποιώντας την εντολή:

```
ARCH=arm make savedefconfig
```

Μετά, αντιγράψτε το πρόσφατα δημιουργημένο defconfig αρχείο στον κατάλογο **arch/arm/configs** με όνομα android_devkit8000_defconfig .

Μόλις τελειώσετε και με αυτό, αλλάξτε το αρχείο με τον ορισμό των επιλογών του BeagleBoard έτσι ώστε να χρησιμοποιεί το νέο αρχείο.

3.12 Καθορισμός πρόσβασης USB στον σταθμό εργασίας σας

Αν εκτελέσετε μια οποιαδήποτε εντολή αυτή τη στιγμή, το πιο πιθανό είναι να λάβετε ένα μήνυμα λάθους σχετικά με τα δικαιώματα πρόσβασης. Αυτό συμβαίνει διότι η συσκευή USB που σχετίζεται με την πλακέτα **DevKit8000** δεν έχει τα κατάλληλα δικαιώματα χρήσης προκειμένου να σας αφήσει να την ανοίξετε.

Πρέπει να σιγουρευτούμε ότι το udev θέτει τα σωστά δικαιώματα έτσι ώστε να έχουμε πρόσβαση σαν χρήστες. Για να γίνει αυτό, δημιουργήστε το αρχείο **/etc/udev/rules.d/51-android.rules** και αντιγράψτε την γραμμή:

```
SUBSYSTEM=="usb", ATTR{idVendor}=="18d1",
ATTR{idProduct}=="0001", MODE="0600", OWNER="<username>"
```

Τώρα, συνδέστε και αποσυνδέσετε την πλακέτα DevKit8000 και θα πρέπει να μπορείτε να χρησιμοποιείτε το adb από τον χρήστη σας.

Λήψη αρχείων καταγραφής από την συσκευή.

Το ADB παρέχει πολλά χρήσιμα χαρακτηριστικά για να δουλέψετε με μια ήδη υπάρχουσα συσκευή Android. Το πρώτο πράγμα που θα δούμε είναι πως παίρνουμε τα αρχεία καταγραφή από ολόκληρο το σύστημα. Για να το καταφέρετε, απλά τρέξτε:

```
adb logcat
```

Θα δείτε τις καταγραφές της συσκευής στο τερματικό σας. Αυτή είναι μια τεράστια ποσότητα πληροφοριών και είναι αρκετά δύσκολο να βρείτε αυτό που ψάχνετε.

Το πρώτο πράγμα που μπορούμε να κάνουμε είναι να κατεβάσουμε έναν μικρό wrapper για να μας παρέχει χρωματιστά τις καταγραφές.

Μπορείτε να το βρείτε εδώ:

http:// jsharkey.org/downloads/coloredlogcat.pytxt

Μόλις το κατεβάσετε, απλά τρέξτε το και θα δείτε τις καταγραφές χρωματιστά και σε ένα εύκολο φορμάτ στοίχισης για ανάγνωσης.

Το ADB επίσης παρέχει κάποια φίλτρα για μια πιο καθαρή έξοδο.

Αυτά στοιχίζονται με το Tag:Priority συντακτικό.

Για παράδειγμα, αν θέλετε όλες τις καταγραφές από το MyApp, απλά δώστε

adb logcat MyApp:*

Παραδοτέο C5: Screenshot στο οποίο να φαίνονται οι καταγραφές από το MyApp.

Χρησιμοποίηση κελύφους σε μια συσκευή

Η ύπαρξη του κελύφους σε μια συσκευή μπορεί να αποδειχθεί αρκετά χρήσιμη. Το ADB παρέχει τέτοια χαρακτηριστικά αν και το ενσωματωμένο κέλυφος είναι αρκετά στοιχειώδες. Για να έχετε πρόσβαση στο κέλυφος, απλά δώστε:

```
adb shell
```

Μπορείτε επίσης να τρέξετε μια εντολή απευθείας στη συσκευή χάρη στο adb shell. Για να το κάνετε αυτό, απλά προσαρτήστε την εντολή στο τέλος. Τώρα, προσπαθήστε να πάρετε όλα τα ενεργά συστήματα αρχείων από την συσκευή.

3.13 Εντολές αρχείων Push/Pull στη συσκεύη

Με την ίδια λογική, το **ADB** σας επιτρέπει να ανακτήσετε αρχεία από την συνδεδεμένη συσκευή και να τα στείλετε σε αυτό, χρησιμοποιώντας τις εντολές **push και pull**.