

Bermain-main Dengan Simputer dan Embedded Linux

By [Pramode C.E](#)

1. Bermain-main Dengan Simputer dan Embedded Linux

Simputer adalah peralatan handheld berbasis CPU StrongArm yang menggunakan Linux. Aslinya didesain oleh para profesor di Indian Institute of Science, Bangalore. Peralatan ini bertujuan sosial untuk menghadirkan komputerisasi dan konektivitas di dalam komunitas rural. Artikel ini akan memberikan tutorial pengenalan pemrograman Simputer (dan peralatan handheld berbasis ARM sejenis - ada banyak sekali di pasaran). Pembaca diharapkan memiliki pengalaman pemrograman Linux. Peringatan: Saya mencoba untuk menjelaskan apa yang telah saya lakukan pada Simputer, dan saya melakukannya tanpa mengalami permasalahan - jika anda mengikuti instruksi saya dan hasilnya adalah handheld anda menjadi berasap - saya sama sekali tidak bertanggung jawab!

1.1. Perangkat Keras/Lunak

Peralatan ini dilengkapi dengan sebuah CPU Intel StrongArm (SA-1110). Ukuran flash memory yang digunakan adalah 32MB atau 16MB dan RAM sebesar 64MB atau 32MB. Fitur-fiturnya meliputi:

1. USB master, dan juga port slave.
2. Serial port standar.
3. Infra Red communication port.
4. Smart card reader.

Beberapa dari fitur di atas dapat diperoleh dengan menggunakan sebuah 'docking cradle' yang disediakan. Sumber daya dapat diperoleh dari baterai isi ulang atau saluran AC external.

Simputer menggunakan GNU/Linux - kernel versi 2.4.18 (dengan beberapa patch) bekerja dengan baik. Juga dilengkapi dengan binary untuk sistem X-Window dan beberapa program utilitas sederhana. Lebih lengkapnya dapat diperoleh dari situs proyek tersebut di www.simputer.org.

1.2. Menghidupkan Simputer

Untuk menghidupkannya, tidak lebih dari sekedar menekan 'tombol power'. Anda akan melihat munculnya gambar kecil tux dan dalam beberapa detik, anda akan mendapatkan X. Layar LCD yang digunakan adalah *touch sensitive* dan anda bisa menggunakan 'stylus' kecil (para *geek* menggunakan kuku jari!) untuk memilih aplikasi dan menjelajahi antarmuka grafis. Jika anda ingin menggunakan input keyboard, bersiap-siaplah untuk menderita dengan melakukan manipulasi menggunakan stylus dan sebuah 'soft keyboard' yang tidak lain adalah sebuah program GUI dimana anda bisa memilih alfabet atau simbol-simbol yang lain.

1.3. Menunggu munculnya bash

GUI adalah untuk anak-anak. Anda tidak akan merasa puas sampai anda melihat prompt bash tua yang sudah terpercaya. Baiklah, anda tidak harus banyak berusaha. Simputer

memiliki sebuah port serial - sambungkan kabel serial yang telah disediakan ke port - hubungkan ujung lainnya ke port yang bebas pada PC Linux host anda (dalam kasus saya, /dev/ttyS1). Sekarang jalankan sebuah program komunikasi (saya menggunakan 'minicom') - anda harus mengkonfigurasi program tersebut sehingga ia menggunakan /dev/ttyS1 dengan kecepatan komunikasi 115200 (demikianlah apa yang dikatakan manual Simputer - jika anda menggunakan handheld lain yang mirip, angka ini tidak harus sama persis) dan format 8N1, serta *flow control* perangkat keras dan lunak dalam keadaan disabled. Melakukan hal ini dengan minicom adalah sangat mudah - gunakan perintah:

```
minicom -m -s
```

Jika konfigurasinya telah selesai - ketikkan:

```
minicom -m
```

dan bersiaplah untuk mendapatkan kejutan. Anda akan segera melihat sebuah prompt login. Anda seharusnya sudah bisa mulai menyetikkan user name/password dan log on. Anda seharusnya sudah bisa menjalankan perintah-perintah sederhana seperti 'ls', 'ps' dan sebagainya - anda bahkan sudah bisa menggunakan 'vi'.

Jika anda tidak terbiasa menjalankan program komunikasi pada Linux, anda mungkin merasa penasaran apa yang sebenarnya terjadi. Tidak ada yang istimewa - ini hanyalah keajaiban Unix yang biasa. Sebuah program yang berada pada Simputer mengawasi port serial (port serial Simputer, dinamakan ttySA0) - ketika anda menjalankan minicom pada PC Linux, anda membuat sebuah hubungan dengan program tersebut, yang mengirimkan prompt login kepada anda melalui kabel serial, membaca respon anda, mengotentifikasi diri anda dan memunculkan sebuah shell dimana anda bisa berinteraksi melalui kabel.

Jika minicom telah menginisialisasi port serial pada PC, anda dapat membuat 'script' interaksi anda dengan Simputer. Anda akan memanfaatkan ide bahwa program yang berjalan pada Simputer selalu mengawasi data yang melewati kabel serial - program tersebut tidak peduli apakah data berasal dari minicom itu sendiri ataukah sebuah script. Anda bisa mencoba eksperimen berikut ini:

1. Bukalah dua buah console (pada PC Linux)
2. Jalankan minicom pada satu console, log on ke Simputer
3. Pada console yang lain, ketikkan `echo ls > /dev/ttyS1`
4. Beralihlah ke console yang pertama - anda akan melihat bahwa perintah `ls` telah dieksekusi pada Simputer.

1.4. Membuat jaringan USB

Simputer didistribusikan bersama sebuah port slave USB. Anda dapat membuat sebuah hubungan TCP/IP antara PC Linux dan Simputer melalui antarmuka USB ini. Inilah langkah-langkah yang harus anda perhatikan:

1. Pastikan anda memiliki distribusi Linux yang cukup baru - Red Hat 7.3 sudah mencukupi.

2. Masukkan salah satu ujung kabel USB ke slot slave USB Simputer, kemudian boot Simputer.
3. Boot PC Linux anda. JANGAN dulu menghubungkan ujung yang lain dari kabel USB ke PC. Login sebagai root pada PC.
4. Jalankan perintah 'insmod usbnet' untuk memuat modul kernel yang mengaktifkan dukungan jaringan USB pada PC Linux. Pastikan bahwa modul tersebut telah termuat dengan menjalankan 'lsmod'.
5. Sekarang masukkan ujung lain dari kabel USB ke slot USB yang bebas pada PC Linux. Subsistem USB pada kernel Linux seharusnya dapat langsung mengenali peralatan yang dihubungkan. Pada PC Linux saya, segera setelah memasukkan kabel USB, saya mendapatkan pesan kernel sebagai berikut (yang dapat dilihat dengan menjalankan perintah 'dmesg'):

```
usb.c: registered new driver usbnet
hub.c: USB new device connect on bus1/1, assigned device
number 3
usb.c: ignoring set_interface for dev 3, iface 0, alt 0
usb0: register usbnet 001/003, Linux Device
```

Sampai sejauh ini, anda harus menjalankan beberapa perintah:

1. Jalankan `ifconfig usb0 192.9.200.1' - ini akan menentukan sebuah alamat IP bagi antarmuka USB pada PC Linux.
2. Dengan menggunakan `minicom' dan kabel serial yang disertakan pada paket, login ke Simputer sebagai root. Kemudian jalankan perintah `ifconfig usb0 192.9.200.2' pada Simputer.
3. Cobalah `ping 192.9.200.2' pada PC Linux. Jika anda melihat paket-paket ping berjalan, selamat... Anda telah membuat sebuah hubungan TCP/IP!

Sekarang anda dapat melakukan telnet/ftp ke Simputer melalui hubungan TCP/IP ini.

1.5. Hello, Simputer

Sekarang waktunya untuk memulai pekerjaan yang sesungguhnya. C compiler yang anda gunakan (gcc) secara normal akan menghasilkan kode 'asli', yaitu, kode yang berjalan pada mikroprosesor dimana gcc itu sendiri berjalan - kebanyakan, pada CPU Intel (atau sejenisnya). Jika anda ingin agar program anda berjalan pada Simputer (yang berbasis pada mikroprosesor StrongArm), kode mesin yang dihasilkan oleh gcc harus bisa dimengerti oleh CPU StrongArm - gcc anda haruslah berperan sebagai sebuah cross compiler. Jika anda mendownload kode sumber gcc (lebih baik 2.95.2) bersama dengan 'binutils', seharusnya anda sudah bisa mengkonfigurasi dan meng-compile-nya dengan suatu cara seperti yang anda lakukan pada sebuah cross compiler (yang bisa dilakukan dengan, katakanlah, arm-linux-gcc). Ini mungkin sedikit rumit jika anda melakukannya untuk pertama kali - produsen handheld anda seharusnya juga memberikan CD yang berisi tool-tool yang dibutuhkan dalam bentuk precompiled - sebaiknya anda menggunakannya (tetapi jika anda serius dalam pengembangan embedded, anda harus berusaha mendownload tool-tool tersebut dan membangunnya sendiri).

Dengan berasumsi bahwa anda memiliki arm-linux-gcc yang bisa berjalan dengan baik, anda dapat menulis sebuah program 'Hello, Simputer' yang sederhana, meng-compile-nya menjadi 'a.out', mentransfer-nya lewat ftp ke Simputer dan mengeksekusinya (akan lebih

baik jika anda memiliki satu console pada PC Linux yang menjalankan ftp dan console yang lain menjalankan telnet - segera setelah mengcompile kode, anda dapat mengupload dan menjalankannya dari console telnet - catat bahwa anda mungkin harus memberikan ijin eksekusi kepada kode yang ditransfer lewat ftp dengan menjalankan 'chmod u+x a.out' pada Simputer).

1.6. Sebuah catatan mengenai kernel Arm Linux

Kernel Linux begitu portabel - semua dependensi mesin diisolasi pada direktori-direktori di bawah subdirektori 'arch' (yang berada langsung di bawah root kode sumber kernel, misalnya, /usr/src/linux). Anda akan mendapatkan sebuah direktori yang disebut 'arm' dibawah 'arch'. Ini adalah direktori yang berisi kode khusus untuk CPU ARM untuk kernel Linux.

Porting Linux ke ARM dimulai oleh Russell King. Arsitektur ARM sangat populer di dunia embedded dan ada banyak sekali mesin-mesin yang berbeda dengan nama-nama yang fantastis seperti Itsy, Assabet, Lart, Shannon dan sebagainya, yang semuanya menggunakan CPU StrongArm (sepertinya ada juga jenis CPU ARM yang lain - yang menimbulkan kesimpang siuran). Ada perbedaan kecil pada arsitektur yang digunakan mesin-mesin tersebut sehingga perlu dilakukan 'machine specific tweaks' agar kernel bisa bekerja pada mesin-mesin tersebut. Tweak untuk kebanyakan mesin tersedia pada kernel standar itu sendiri, dan anda hanya tinggal memilih jenis mesin yang digunakan pada saat konfigurasi kernel agar semuanya dapat berjalan dengan baik. Namun yang membuat Simputer sedikit membingungkan, tampaknya tweak untuk spesifikasi khusus Simputer akan menghasilkan kode kernel ARM - tetapi perusahaan yang membuat dan memasarkan peralatan tersebut sepertinya membuatnya dengan spesifikasi yang telah dimodifikasi - dan patch yang dibutuhkan untuk membuat kernel ARM berjalan pada konfigurasi yang telah dimodifikasi tersebut belum terintegrasi ke dalam kernel utama. Namun ini bukanlah suatu permasalahan yang berarti, karena perusahaan pembuatnya akan memasarkan Simputer beserta patch-nya - dan mereka mungkin akan segera merilis official kernel.

1.7. Mendapatkan dan membangun kode kernel

Anda dapat mendownload kode sumber kernel 2.4.18 dari ftp mirror kernel Linux terdekat. Anda akan membutuhkan file 'patch-2.4.18-rmk4' (yang bisa anda dapatkan dari situs FTP ARM Linux ftp.arm.linux.org.uk). Anda mungkin juga membutuhkan patch keluaran perusahaan pembuatnya, katakanlah, 'patch-2.4.18-rmk4-vendorstring'. Diasumsikan bahwa semua file di atas disalin ke direktori /usr/local/src.

1. Pertama, untar distribusi kernel utama dengan melakukan `tar xvfz kernel-2.4.18.tar.gz`
2. Anda akan mendapatkan sebuah direktori yang dinamakan `linux`. Beralihlah ke direktori tersebut dan jalankan `patch -p1 < ../patch-2.4.18-rmk4`.
3. Sekarang jalankan patch yang dikeluarkan perusahaan pembuat. Lakukan `patch -p1 < ../patch-2.4.18-rmk4-vendorstring`.

Sekarang, kernel anda telah siap untuk dikonfigurasi dan dibangun. Namun sebelumnya, anda harus memeriksa Makefile (di bawah /usr/local/src/linux) dan melakukan dua perubahan - akan ada sebuah baris dengan bentuk sebagai berikut:

ARCH := <lots-of-stuff>

dekat dengan baris atas. Ubahlah menjadi:

ARCH := arm

Anda harus melakukan satu perubahan lagi. Perhatikan bahwa Makefile mendefinisikan:

```
AS = ($CROSS_COMPILE)as  
LD = ($CROSS_COMPILE)ld  
CC = ($CROSS_COMPILE)gcc
```

Catat bahwa simbol CROSS_COMPILE disamakan dengan string kosong. Pada kompilasi normal, hal ini akan menghasilkan AS didefinisikan sebagai 'as', CC didefinisikan sebagai 'gcc' dan seterusnya seperti apa yang kita inginkan. Tetapi ketika kita melakukan cross compiling, kita menggunakan arm-linux-gcc, arm-linux-ld, arm-linux-as dan sebagainya. Sehingga anda harus menyamakan CROSS_COMPILE dengan string arm-linux-, dengan cara, pada Makefile, anda harus memasukkan:

```
CROSS_COMPILE = arm-linux-
```

Jika perubahan tersebut telah dimasukkan pada Makefile, anda dapat segera mulai mengkonfigurasi kernel dengan menjalankan 'make menuconfig' (catat bahwa dimungkinkan untuk melakukannya tanpa memodifikasi Makefile, dengan menjalankan 'make menuconfig ARCH=arm'). Mungkin akan membutuhkan sedikit tweak sebelum anda bisa membangun kernel tanpa kesalahan. Anda tidak harus melakukan banyak memodifikasi - defaultnya seharusnya sudah cukup.

1. Anda harus mengubah jenis sistem menjadi sistem ARM berbasis SA1100 dan kemudian memilih implementasi SA11x0 untuk menjadi 'Simputer(Clr)' (atau yang lain, tergantung pada mesin anda). Saya juga meng-enable dukungan fungsi USB SA1100, dukungan USB net link SA11x0 dan SA11x0 USB char device emulation.
2. Dibawah Character devices->Serial drivers, saya mengaktifkan dukungan port serial SA1100, console pada dukungan port serial dan mengeset baud rate default menjadi 115200 (anda mungkin harus mengeset angka yang berbeda pada mesin anda).
3. Di bawah Character devices, SA1100 real time clock dan Simputer real time clock di-enable.
4. Di bawah Console drivers, VGA Text console di-disable.
5. Di bawah General Setup, perintah default kernel diset menjadi `root=/dev/mtdblock2 quite'. Ini mungkin akan berbeda pada mesin anda.

Jika proses konfigurasi telah selesai, anda dapat menjalankan

```
make zImage
```

dan dalam beberapa menit, anda akan mendapatkan sebuah file yang dinamakan 'zImage' di bawah arch/arm/boot. Ini adalah kernel baru anda.

1.8. Menjalankan kernel baru

Saya akan menjelaskan cara termudah untuk menjalankan kernel yang baru.

Seperti jika anda menggunakan LILO atau Grub sebagai boot loader untuk PC Linux anda, handheld juga menggunakan bootloader yang disimpan pada *non volatile memory*. Pada kasus Simputer, bootloader ini disebut sebagai 'blob' (yang saya artikan sebagai boot loader yang dikembangkan untuk Linux Advanced Radio Terminal Project, 'Lart'). Segera setelah anda menghidupkan mesin tersebut, boot loader mulai berjalan - jika anda menjalankan minicom pada PC Linux anda, teruslah menekan tombol 'enter' dan kemudian hidupkan Simputer, bootloader tidak akan melanjutkan boot kernel yang tersimpan pada flash memory, namun ia akan mulai berinteraksi dengan anda melalui prompt yang terlihat seperti berikut:

```
blob>
```

Pada prompt bootloader, ketiklah:

```
blob> download kernel
```

yang mengakibatkan blob menunggu anda mengirimkan kernel image yang di-uuencode melewati port serial. Sekarang, pada PC Linux, anda harus menjalankan perintah:

```
uuencode zImage /dev/stdout > /dev/ttyS1
```

Ini akan mengirimkan kernel image yang di-uuencode melalui COM port - yang akan dibaca dan disimpan oleh bootloader pada RAM Simputer. Jika proses ini telah selesai, kembalilah pada prompt bootloader. Ketikkan:

```
blob> boot
```

dan bootloader akan menjalankan kernel yang telah anda compile dan download.

1.9. Sedikit mengenai kernel hacking

Apa bagusnya peralatan yang baru jika anda tidak dapat melakukan sedikit kernel hacking? Langkah yang saya lakukan setelah mengcompile dan menjalankan kernel yang baru adalah mengecek bagaimana mengcompile dan menjalankan modul kernel. Ini adalah program sederhana yang dinamakan 'a.c':

```
#include <linux/module.h>
#include <linux/init.h>

/* Just a simple module */

int
init_module(void)
{
    printk("loading module...\n");
    return 0;
}
```

```
}  
  
void  
cleanup_module(void)  
{  
    printk("cleaning up ... \n");  
}
```

Anda harus mencompilanya dengan menggunakan perintah:

```
arm-linux-gcc -c -O -DMODULE -D__KERNEL__ a.c  
-I/usr/local/src/linux-2.4.18/include
```

Anda dapat mentransfer file 'a.o' hasil kompilasi melalui ftp ke Simputer dan memuatnya ke dalam kernel dengan menjalankan perintah:

```
insmod ./a.o
```

Anda dapat menghapus modul tersebut dari kernel dengan menjalankan:

```
rmmod a
```

1.10. Menangani Interrupts

Setelah menjalankan program di atas, saya mulai mempelajari kode sumber kernel untuk mengidentifikasi segmen kode yang paling sederhana yang dapat mendemonstrasikan suatu akses pada hardware - dan saya menemukannya pada driver hard key. Simputer memiliki tombol-tombol kecil yang jika ditekan bertindak sebagai tombol panah - tombol-tombol tersebut kelihatannya dihubungkan ke I/O pins general purpose pada CPU ARM (yang juga dapat diatur sebagai sumber interrupt - jika ingatan saya saat membaca manual StrongArm benar). Menulis sebuah modul kernel yang merespon ketika tombol-tombol tersebut ditekan adalah sangat mudah - ini adalah program kecil yang hanya merupakan versi modifikasi dan penyederhanaan dari driver hardkey - anda menekan tombol yang sesuai dengan arah tanda panah - sebuah interrupt kemudian dibangkitkan sehingga menyebabkan pengendali dieksekusi. Pengendali ini hanya akan menampilkan sebuah pesan dan tidak melakukan sesuatu yang lain. Sebelum memasukkan modul, kita harus memastikan bahwa kernel yang berjalan pada Simputer tidak memasukkan kode driver tombol default - anda cukup melihat pada /proc/interrupts.

Compile program di bawah ini menjadi file object (seperti yang kita lakukan pada program sebelumnya), muatlah ke dalam kernel dengan menggunakan 'insmod', lihat /proc/interrupts untuk memeriksa apakah baris interrupt telah didapatkan. Menekan tombol harus berakibat pemanggilan pengendali tersebut - jumlah interrupt yang ditampilkan pada /proc/interrupt juga harus berubah.

```
#include <linux/module.h>  
#include <linux/ioport.h>  
#include <linux/sched.h>  
#include <asm-arm/irq.h>
```

```
#include <asm/io.h>

static void
key_handler(int irq, void *dev_id, struct pt_regs *regs)
{
    printk("IRQ %d called\n", irq);
}

static int
init_module(void)
{
    unsigned int res = 0;
    printk("Hai, Key getting ready\n");
    set_GPIO_IRQ_edge(GPIO_GPIO12, GPIO_FALLING_EDGE);
    res = request_irq(IRQ_GPIO12, key_handler, SA_INTERRUPT,
"Right Arrow Key", NULL);
    if(res) {
        printk("Could Not Register irq %d\n", IRQ_GPIO12);
        return res;
    }
    return res ;
}

static void
cleanup_module(void)
{
    printk("cleanup called\n");
    free_irq(IRQ_GPIO12, NULL);
}
}
```

1.11. Rencana di Masa Depan

Sebuah handheld berbasis Linux memberikan banyak kesempatan untuk melakukan permainan yang serius - semakin saya belajar lebih banyak mengenai peralatan ini, saya harus berusaha untuk berbagi penemuan dengan para pembaca.

1.12. Referensi

1. [Simputer Project](#) Home Page.
2. [Simputerland](#) dan [PicoPeta](#) - informasi mengenai aktivitas pengembangan Simputer dari perusahaan yang memproduksi dan memasarkan produk ini.
3. [Arm Linux](#) Project Home Page
4. [Lart Project](#) Home Page. Ada banyak hal yang menarik di sini. Anda mungkin ingin mencoba link 'Clock Scaling' pada situs ini. Clock scaling mengijjinkan anda untuk mengubah kecepatan clock secara langsung pada prosesor yang sedang berjalan - berguna untuk menghemat baterai.

Saya adalah seorang instruktur yang bekerja untuk IC Software di Kerala, India. Saya senang menjadi seorang ahli kimia organik, namun saya juga melakukan pekerjaan yang mungkin menjadi hal terbaik kedua, yaitu bermain dengan Linux dan mengajar pemrograman!



Copyright © 2003, Pramode C.E. Copying license

<http://www.linuxgazette.com/copying.html>

Published in Issue 87 of *Linux Gazette*, February 2003

Diterjemahkan oleh [Triyan W. Nugroho](#)