APLIKASI GPIO (GENERAL PURPOSE INPUT/OUTPUT) PADA PLATFORM PROSESOR ARM11 BERBASIS SISTEM EMBEDDED LINUX

Ferdian Nasruddin[(1)], A.R. Anom Besari, S.ST[(2)], Fernando Ardilla, S.ST, M.T[(2)]
[(1)]Mahasiswa Program Studi Teknik Komputer, [(2)]Dosen Program Studi Teknik Komputer
Politeknik Elektronika Negeri Surabaya – Institut Teknologi Sepuluh Nopember (ITS) Surabaya
Kampus ITS, Sukolilo, Surabaya 60111

ABSTRAK

Kernel Linux saat ini telah mendukung banyak *platform* prosesor. Diantaranya adalah prosesor dengan arsitektur ARM yang telah banyak digunakan untuk *chip CPU* pada perangkat *mobile* seperti *handphone* atau PDA. *System on chip* pada boardphyCORE-i.MX31 dengan *core* prosesor ARM1136JF-S memiliki banyak fitur yang bisa dikembangkan untuk membuat *prototype* perangkat *embedded*. Dalam proyek akhir ini dipelajari tentang bagaimana mengembangkan sebuah perangkat *embedded* berbasis sistem *embedded* Linux. Dipilihnya Linux sebagai sistem operasi tidak lepas dari salah satu kelebihannya yakni *open source*, atau terbukanya kode sumber sehingga para pemula dan *developer* bisa mempelajari bahkan berkontribusi untuk mengembangkan sistem agar lebih baik. Dalam prakteknya akan dibahas tentang bagaimana langkah yang harus dikerjakan mulai dari mempersiapkan host untuk lingkungan pengembangan sistem dan bagaimana memanfaatkan periferal yang ada pada target yakni *board* phyCORE-i.MX31. Sebagai bahan uji akan dipelajari tentang *GPIO* pada *board* tersebut, bagaimana menambahkan *driver*, memodifikasi, mengkompilasi program dan membangun *kernel* serta bagaimana menanamkan sistem Linux ke dalam *board* tersebut. Pengujian untuk langkah awal pengembangan sampai menanamkan *kernel* baru kedalam board telah berhasil, namun untuk uji pengaksesan *GPIO* untuk *board* ekspansi pada proyek akhir ini masih belum menemukan solusi. Hal ini dimugkinkan karena kesalahan dalam penerapan metode pengaksesan *GPIO* pada *board* ekspansi atau karena masih sangat terbatasnya pengetahuan tentang *GPIO* pada Linux oleh peneliti. Harapannya, dengan proyek akhir ini akan sedikit membuka wawasan untuk lebih dikembangkannya sistem Linux untuk perangkat embedded di Indonesia dan pada lingkungan pendidikan khususnya.

Kata kunci: *Embedded devices, Linux, phyCORE-i.MX31, GPIO.*

1. PENDAHULUAN

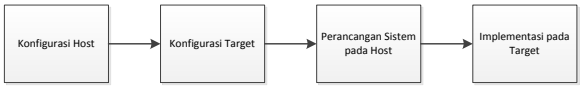
Pada umumnya sistem benam (*embedded system*) yang menggunakan fungsi sederhana tidak membutuhkan sistem operasi. Misalnya pada sistem berbasis mikrokontroler seri MCS51, yang sudah mampu merealisasikan *close-loop* control hanya dengan *software*. Akan tetapi semakin berkembangnya teknologi prosesor, desain sistem benam menjadi semakin rumit seperti mengontrol sebuah peralatan atau memonitor kinerja dan tugas sebuah mesin, penjadwalan multitask untuk sistem *real-time* dan *multitasking* dalam sistem komunikasi. Akibatnya untuk menerapkan desain metode tersebut pada sebuah sistem akan menjadi sulit. Jadi pada sistem benam yang rumit dibutuhkan sebuah sistem operasi yang mampu mengatasinya.

Pada proyek akhir ini dipilih Linux sebagai sistem operasi karena dukungannya terhadap banyak platform prosesor. Sifatnya yang open source menjadikan lebih mudah dalam pengembangannya. Sistemembedded Linux meskipun telah ditulis ulang dan ukurannya lebih kecil namun masih memiliki kelebihan dari Linux standar seperti: stabilitas, transplantasi *API* yang bagus untuk banyak *platform*, fungsi jaringan yang sangat baik, mendukung berbagai macam *system* dokumen, dan lain-lain.

Dalam proyek akhir ini ditujukan kepada sebuah penelitian tentang bagaimana mengembangkan sebuah sistem Linux ke dalam sebuah board *development* dengan arsitektur *CPU* tertentu. Kemudian membuat aplikasi didalamnya yang berupa *software* maupun aplikasi berupa *hardware*. Digunakan phyCORE-i.MX31 dengan *core* proseseor ARM1136JF-S dengan kecepatan prosesor 532MHz (maks) untuk penelitian ini. *Board* ini cukup untuk dikembangkan sebagai perangkat multimedia dan komunikasi karena banyaknya periferal yang mendukung seperti *LCD touch, sound card, usb host controller* dan sebagainya.

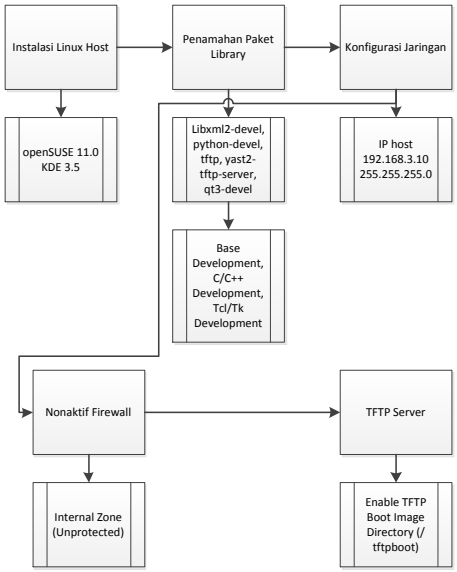
2. PERANCANGAN SISTEM

Pada perencanaan dan pembuatan sistem ini akan dibahas tentang proses instalasi dan konfigurasi perangkat lunak agar dapat digunakan untuk mengembangkan aplikasi didalam *board kit* phyCORE-i.MX31. Secara keseluruhan sistem *software* akan dikembangkan didalam *host*, dimana *host* disini adalah sebuah *Personal Computer (PC)* dengan spesifikasi minimum yang dibutuhkan. Dengan menggunakan kabel serial *DB-9* dan kabel *Ethernet Cross-over*, host dihubungkan dengan *target (*phyCORE *development board)*.



Gambar 1. Blok Diagram Perancangan Umum

2.1. Instalasi dan Konfigurasi *Software* Pada *Platform Host*



Gambar 2. Blok diagram konfigurasi software pada host

1) Instalasi Paket Software
   Dibutuhkan beberapa *library*, paket *software* serta *development tool* yang secara *default* pada OpenSUSE 11.0 belum ter-*install*. Paket yang dibutuhkan diantaranya: *libxml2-devel; python-devel; tftp* dan *qt3-devel*.
2) Konfigurasi Kartu Jaringan
   Untuk dapat bertukar data antara *host* dengan *target*, selain menginstal *TFTP server* perlu diberikan beberapa *setting* terhadap kartu jaringan. Untuk itu diperlukan konfigurasi alamat *IP* untuk *host*. Secara default *IP* dan *subnet mask* dari target adalah *192.168.3.11/ 255.255.255.0*.
3) Me-*non*-aktifkan *Firewall*
   Untuk masalah dengan koneksi pada *target*, *firewall* harus di-*non*-aktifkan.
4) *SetUp TFTP Server*
   Agar *image kernel* dapat di-*download*-kan ke dalam *target*, maka *TFTP* dijalankan terlebih dahulu.
5) *Setup* Linux-i.MX31-Kit
   Setup ini akan meng-*install program-progam* berikut:
   a) *GNU C/C++ cross development toolchain* - *toolchain* ini digunakan untuk mengembangkan program pada *target* menuju *host*.
   b) Eclipse *SDK* dengan *CDT* - Eclipse *SDK* adalah *platform* dan *framework* aplikasi yang mana dapat menggunakan *GNU C/C++ cross development toolchain* untuk membangun perangkat lunak.
   c) *Microcom* - program untuk kominikasi *serial* dengan *target*.
   d) Linux *Kernel archive* - arsip *kernel* ini berisi *source code kernel* Linux serta semua *patch* yang dibutuhkan untuk mengkompilasi *kernel* pada phyCORE-i.MX31.
   e) *HelloWorld* - contoh *program* ini dapat digunakan untuk menguji cara men-*download* dan menjalankan program pada *target*.
   f) *mkimage* - *program* ini akan digunakan untuk membuat *file kernel image* untuk *target*.

6) Konfigurasi dan Kompilasi *Kernel*
   Kernel biasanya akan dibangun untuk arsitektur mesin asli dari *host* (*PC*). Untuk menggunakan arsitektur ARM dan ARM *cross compiler* yang cocok untuk phyCORE-i.MX31, sebagai gantinya harus ditentukan arsitektur dan *cross compiler* pada *command line*. Dalam proyek akhir ini digunakan paket *kernel* Linux yang telah disediakan oleh PHYTEC yang paketnya sudah dalam kofigurasi *default* untuk arsitektur dan *cross compiler* yang digunakan yakni dengan kode PCM037. Adapun untuk versi *kernel* yang digunakan adalah versi *kernel 2.6.x*.

```
make xconfig ARCH=arm
```

   Perintah ini digunakan untuk menjalankan program *qconf* (*tool* untuk mengkonfigurasi *kernel*). Parameter *xconfig* `ARC=arm` untuk prosesor dengan arsitektur ARM.

7) Instalasi *Boot Loader*
   Boot loader yang digunakan untuk phyCORE-.iMX31 adalah *U-Boot (Universal Boot Loader)*. Instalasi *boot loader* ini menggunakan *software* AdvancedToolKit buatan *Freescale* yang berjalan pada Micosoft Windows. *Boot loader* ini berfungsi untuk me-*load* sistem operasi Linux.
8) Konfigurasi *U-Boot Environment Variables*
   Agar *board* dapat *booting* ke dalam sistem Linux dengan benar, maka perlu di *setting environment* variable untuk *U-boot*. Didalamnya dapat di-setting *IP address, net mask*, nama *file kernel, rootfs, u-boot* dan sebagainya, untuk parameter *booting* pada sistem Linux yang benar.
9) Membuat Lingkungan Kerja
   Pada bagian ini akan dibuat lingkungan kerja untuk pengembangan Linux, yakni membangun *Board Support Package*. *Board Support Package (BSP)* ini berisikan banyak

tool, paket dan konfigurasi untuk mengembangkan sistem operasi Linux yang digunakan sesuai dengan kebutuhan dan *hardware* yang ada pada *board* phyCORE-i.MX31.

10) Membangun *root filesytem* phyCORE-i.MX31
    Yang dikerjakan pada bagian ini diantaranya adalah:
    a) Membangun *Toolchain*
    b) Memilih *Platform Software*
    c) Memilih *Platform Hardware*
    d) Mengkonfigurasi *Toolchain*
    e) Membangun image file.

11) Pada proses membangun *root filesystem* dan perubahan *kernel* akan terjadi perubahan pula pada konfigurasi *u-boot*. Maka sebelum Linux dijalankan perlu di-*update* konfigurasi *U-Boot*-nya.

3. HASIL PENGUJIAN

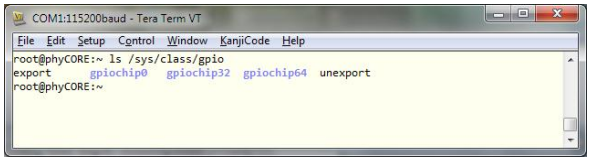
Pada uraian ini akan diuraikan tentang hasil pengujian terhadap hasil konfigurasi beserta uji coba untuk pengaktifan pin *GPIO* pada board ekspansi.

Untuk mengakses *GPIO* pada *board* ekspansi merupakan wilayah *userspace* dimana jalur *I/O* yang ada dapat dimanipulasi secara langsung. Untuk *I/O* yang spesifik, misalnya pada *touchscreen* atau *usb controller* hanya dapat diprogram melalui *kernel space* karena bersentuhan langsung dengan *driver* yang khusus untuk memuat periferal tersebut pada sistem Linux.

Pada *board* phyCORE-i.MX31 tidak dijelaskan secara detail untuk mengakses pin yang ada pada board ekspansi. Berikut contoh cara mengakses *GPIO* dari *userspace* dengan metode *interfacing gpio-sysfs*.

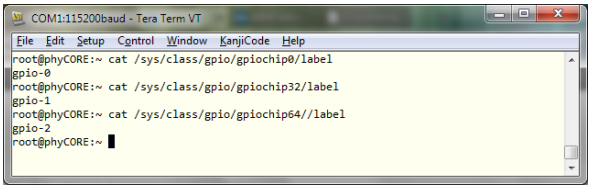
a) Struktur file
Ketik pada terminal:

```
ls /sys/class/gpio
```



Gambar 3. Struktur file *GPIO* pada *phyCORE*

Struktur file tersebut menunjukkan bahwa pada dalam phyCORE-i.MX31 terdapat 3 buah chip yang menyediakan pin GPIO yakni *gpiochip0*, *gpiochip32* dan *gpiochip64*.



Gambar 4. Label dari masing-masing *chip GPIO* pada phyCORE

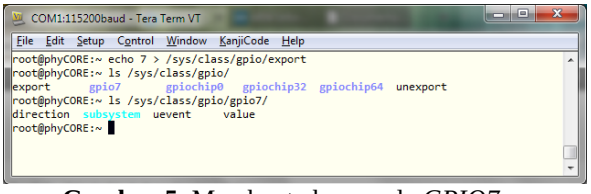
Tiap *chip* direpresentasikan sebagai sebuah *PORT*, misalnya *gpiochip0* adalah *PORT A*, *gpiochip32* adalah *PORT B* dan *gpiochip64* adalah *PORT C*.

b) *GPIO* Number
Untuk mendapatkan akses ke *GPIO*, maka harus dicari nomor *GPIO*. Hal ini dilakukan dengan mencari angka dasar dari chip *GPIO* yang dimiliki kemudian menambahkan jumlah pin pada chip tersebut. Sebagai contoh, PA7 adalah *Port A* ditambah *base 7*.
Setiap base dapat ditemukan dengan melihat pada file "*base*" dari *folder gpiochipN* yang sesuai. Nomor terdapat pada akhir nama *folder*. Jadi, dalam contoh ini dicari *Port A*, yang disebut *gpio-0*. Label ini ditemukan dalam *folder gpiochip0* dan dengan demikian jumlah basis 0. Yang dinginkan adalah pin 7 sehingga 7 + 0 adalah 0, sehingga *gpio number*-nya adalah 7.

c) Mengakses *GPIO*
Kadang-kadang kernel telah memberikan akses ke sebuah *GPIO*. Jika demikian maka dapat dilihat folder *gpioN* pada `/sys/class/gpio` di mana N adalah nomor *GPIO*. Jika tidak terdapat folder ini maka perlu membuatnya. Setelah memiliki nomor *GPIO*, selanjutnya adalah memintanya pada kernel.

```
#: echo 7 >> /sys/class/gpio/export
```

Setelah ini, akan terlihat folder `/sys/class/gpio/gpio7`. Dalam folder ini tampak atribut yang dibutuhkan untuk mengontrol *GPIO* tersebut.

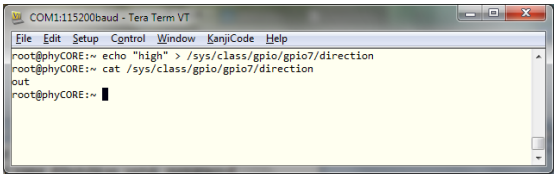


Gambar 5. Membuat akses pada *GPIO7*

d) Men-*setting* directions

Untuk menentukan direction, tulis salah satu dari atribut direction *GPIO* berikut; *"in", "out", "high"* atau *"low"*. *"low"* dan *"out"* memiliki efek yang sama, untuk mengubah pin untuk *output* yang awalnya rendah (*low*). *"high"* merubah pin *output* yang awalnya tinggi. *"in"* merubah pin untuk *input*.

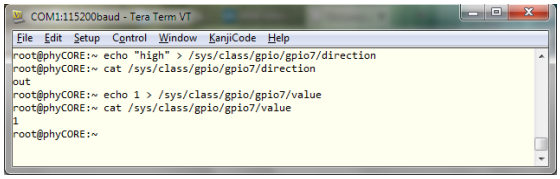
File ini akan dibaca sebagai *"in"* atau *"out"* sehingga dapat menentukan *direction*-nya.



Gambar 6. Men-*setting direction* pada *gpio7*

e) Mendapatkan *state*

Untuk memberikan state *"1"* atau *"0"* cukup menuliskannya pada *value*.



Gambar 7. Menambahkan *state* pada *value*

Setelah langkah tersebut diatas, diukur pada *board* ekspansi pada pin *GPIO7* dengan menggunakan *avometer*.

Gambar 8 Cek *Output* pada pin *VCC*

Pada pin *VCC* ketika di-cek dengan menggunakan *avometer* menghasilkan tegangan 4.262 *Volt*. Ini berarti pin VCC berfungsi dengan benar.

Ketika pengujian terhadap pin GPIO7 dengan mengunakan *avometer*. Tidak terjadi perubahan apapun, *avometer* menunjukkan angka 0.001 *Volt*.

Gambar 9. Cek *Output* pada pin GPIO7

Hasil dari pengujian kali ini, tidak terdapat perubahan apapun. Sehingga percobaan untuk mengakses *GPIO* pada expansion board belum berhasil dengan metode ini.

4. KESIMPULAN

Dari tugas akhir yang telah dilakukan maka dapat diambil beberapa kesimpulan yaitu:

1. Konfigurasi software untuk host menitik beratkan pada komunikasi antara lingkungan pengembangan sistem dengan board tempat dikembangkan sistem. Seperti konfigurasi perangkat jarigan untuk mentransfer file lintas root dari Linux dan komunikasi serial untuk interfacing dengn sistem Linux pada target.
2. Hal penting yang perlu diketahui dalam perencanaan sistem host adalah pengetahuan tentang command-command pada Linux. Baik untuk shell scripting dan pengetahuan tentang administrasi sistem Linux.
3. Konfigurasi kernel pada board ini masih terbatas pada penambahan dan pengurangan modul-modul yang ada pada board.
4. U-Boot berguna untuk me-load sistem dan menjalankan start-up program yang telah ditanamkan pada target.

DAFTAR PUSTAKA

[1] *Luo, Lei. Da,i Xuefeng. Shi, Yan. Development of Direct Current Motor Control System Based on Embedded Linux System. IEEE Computer Society, 2007.*

[2] *Yuhua, Zhang. Zifei, Zhang. Design of Infrared Power Meter Reading System based on Embedded Linux Operation System. IEEE Computer Society, 2010.*

[3] *McLoughlin, Ian. Aendenroomer, Anton. Linux as a Teaching Aid for Embedded Systems. IEEE Computer Society, 2007*

[4] *Hallinan, Christoper. Embedded Linux Primer Second Edition, A Practical Real-World Approach. Prentice Hall: 2011.*

[5] *Corbet, Jonathan. Rubini, Alessandro. Kroah-Hartman, Greg. Linux Device Drivers Third Edition. O'Reilly: 2005.*
[6] *Yaghmour, Karim. Building Embedded Linux Systems. O'Reilly: 2003*
[7] *The Linux Information Project (2006). http://www.linfo.org/index.html*
[8] *Free Electrons – Embedded Linux Experts (2011). http://free-electrons.com/*
[9] *The Linux Cross Reference (2011). http://lxr.linux.no/+trees*
[10] *Embedded Linux Wiki (2011). http://elinux.org/Main_Page*
[11] *Pengutronix (2011). OSELAS®.Toolchain( ). http://www.pengutronix.de/oselas/toolchain/index_en.html*
[12] *Pengutronix (2011). BSPs for products by Phytec AG. http://www.pengutronix.de/oselas/bsp/phytec/index_en.html*
[13] *The Linux Documentation Project (2011). http://tldp.org/*
[14] *Kernel (2011). GPIO Interfaces. http://www.kernel.org/doc/Documentation/gpio.txt*