Implementasi *Embedded* Linux pada Jaringan Sensor Nirkabel *Platform* Imote2

Fajar Purnama, I Made Oka Widyantara, Nyoman Putra Sastra
Jurusan Teknik Elektro
Fakultas Teknik, Universitas Udayana
Jimbaran, Indonesia
Email: fajar.purnama@yahoo.com, oka.widyantara@unud.ac.id, putra.sastra@unud.ac.id

***Abstrak*—Sebelumnya JSN menggunakan sistem operasi Intel Platform X, SOS, dan TinyOS. Kini Platform X dan SOS tidak dikembangkan lagi, sehingga banyak peneliti menggunakan TinyOS. Pada akhir pengguna TinyOS pada *platform* Imote2 menemukan banyak keterbatasan seperti penerapan pada *routing* yang *complex*. Oleh karena itu komunitas *Embedded* Linux pengembangkan *embedded* Linux untuk *platform* Imote2. Pada makalah ini dibahas secara rinci tahap untuk meng*embed* Linux pada *target* yaitu perangkat JSN radio *sensorboard platform* Imote2. Host merupakan Linux *operating system*. Peng*embed*an meliputi 3 komponen utama yaitu *bootloader*, Linux *kernel*, dan *filesystem*. *Embed* dilakukan dengan proses *flashing* pada JTAG *interface* menggunakan *software* OpenOCD. Setelah proses *embed*, konfigurasi pada *target* melalui koneksi serial. Konfigurasi meliputi pengaktifkan alamat IP, SSHD, dan radio secara otomatis. Terakhir dibandingkan performansi *target* yang menggunakan IEEE 802.11 WLAN dan IEEE 802.15.4 ZigBee sebagai media transmisi. Hasilnya penggunaan IEEE 802.11 WLAN lebih boros terhadap *memory* dan daya listrik.**

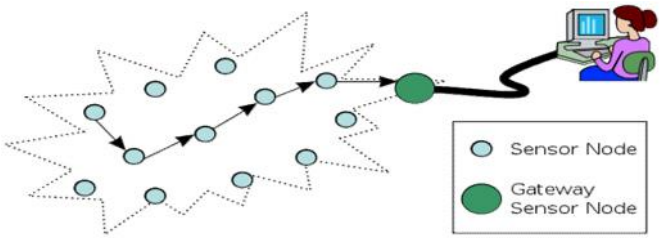
***Kata Kunci*—JSN, Imote2, *Embedded* Linux, ZigBee, WLAN, JTAG, OpenOCD.**

I. PENDAHULUAN

Secara keseluruhan makalah ini terdiri dari 4 bagian. Bagian pertama "Pendahuluan" menjelaskan sekilas mengenai jaringan sensor nirkabel (JSN), beberapa penelitian yang telah dilakukan pada bidang JSN, dan hal yang akan dibahas pada makalah. Bagian kedua "Intel Mote 2 (Imote2)" menjelaskan perangkat JSN dengan *platform* Imote2. Pada bagian ini dapat dilihat bentuk fisik dari alat dan disebutkan beberapa sistem operasi yang diterapkan pada *platform* ini. Pada bagian ketiga "Implementasi *Embedded* Linux" dituliskan langkah-langkah untuk instalasi sistem operasi Linux pada *platform* Imote2. Bagian terakhir adalah "Simpulan".

JSN atau lebih dikenal dengan *wireless sensor network* (WSN) merupakan perangkat sensor yang saling terkomunikasi secara nirkabel. Perangkat-perangkat ini di letakkan pada daerah yang luas secara geografis dan membentuk jaringan sensor. JSN ini tidak harus terhubung dengan Internet. Tugas utama dari sensor ini sensor ini adalah mengumpulkan informasi dari lingkungan sekitar, setelah itu mengirimkan informasi tersebut ke perangkat pengguna melalui jaringan sensor. Perangkat ini telah diterapkan di bidang sipil, medis, dan lain-lain [1].

Adanya JSN berdasarkan adanya banyak keterbatasan, seperti keterbatasan daya dan keterbatasan kemampuan kanal nirkabel. Oleh karena itu sensor nirkabel tersebut membentuk suatu jaringan disebut JSN. Selain diperlukan penempatan sensor secara strategis (menanggulangi *coverage hole*) diperlukan juga upaya untuk menghemat energi dan waktu dalam memberian informasi secara kontinyu karena sumber daya pada JSN terbatas [2]. Gambaran JSN dapat dilihat pada Gambar 1.



Gambar. 1 Gambaran JSN [3]

Pada JSN telah terdapat beberapa penelitian sebelumnya seperti penelitian [4] yang menggunakan banyak perangkat JSN *camera* untuk menangkap citra yang sama. Perangkat tersebut ditaruh pada posisi yang berbeda namun mengarah pada obyek yang sama. Dengan menggabungkan citra yang ditangkap pada sudut yang berbeda dapat meningkatkan kualitas citra. Pada akhir penelitian diusulkan suatu bentuk JSN yang bekerja pada konsep ini. Penelitian [5] mengarah pada pemodelan JSN *camera* dengan konsumsi daya rendah dengan menetapkan kualitas citra. Pada penelitian tersebut menyimpulkan beberapa *point* yang perlu diperhatikan pada pemodelan JSN *camera*, yaitu (i) metode pemilihan kamera (ii) metode dan strategi kompresi citra (iii) metode transmisi citra. Tetapi pada makalah ini tidak akan membahas JSN pada ruang lingkup tersebut.

Tidak seperti pada [4] dan [5] penelitian ini tentang *embedded* Linux pada JSN, seperti yang telah dilakukan oleh Peneliti [6], [7], dan [11]. Pada [6] dan [7], performansi *platform* Imote2 pada jaringan sensor nirkabel masing-masing menggunakan jaringan IEEE 802,15.4 Zigbee dan IEEE 802.11 WLAN. Sedangkan model implementasi embedded linux pada JSN, kedua penelitian ini menggunakan skema yang sama yaitu (i) menghubungkan perangkat JSN multimedia ke komputer (ii) instalasi *bootloader*, Linux *kernel* dan *filesystem* (iii) mengatur jaringan *internet protocol* (IP)

dan *secure shell daemon* (SSHD) (iv) mengaktifkan radio (v) mengukur konsumsi *memory* dan daya listrik.

Setelah hal tersebut dilakukan baru kita dapat menambahkan beberapa fitur, seperti menangkap citra dengan *camera sensor board* IMB400 yang terlihat pada [8] dan [9]. Namun pada makalah ini tidak membahas sejauh itu, tetapi membahas mengenai *embedded* Linux pada *radio sensorboard platform* Imote2 dengan rinci.

II. INTEL MOTE 2 (IMOTE2)

Imote2 merupakan *platform* pada perangkat JSN yang dikembangkan oleh Intel Research pada bagian penelitian Platform X. Perangkat ini dibangun dengan konsumsi daya listrik yang rendah, dengan *processor* PXA271 XScale CPU, dan terintegrasi pada IEEE 802.15.4 ZigBee [10]. *Processor* ini (Intel Xscale *processor* PXA271) dapat beroperasi pada tegangan rendah (0.85V) dan frekuensi 13MHz hingga 104MHz. Frekuensi dapat dinaikkan hingga 416MHz dengan mengatur tegangan [11]. Secara umum Imote2 terdiri dari 4 bagian seperti terlihat pada Gambar 2.

(a) (b) (c) (d)

Gambar. 2 (a) radio *processor board* (IPR2400) [11] (b) *interface board* [11] (c) *sensor board* (IMB400) [8] (d) *power supply board* (IBB2400) [12].

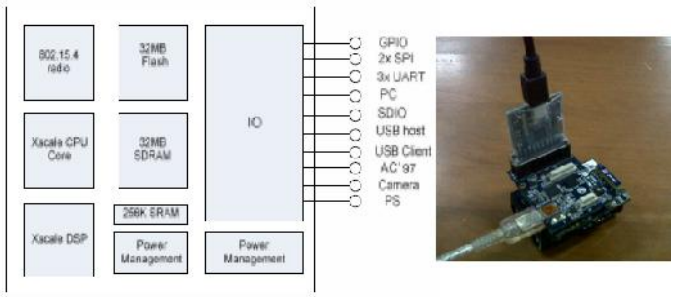
PXA271 terdiri dari 3 *chip* (i) *processor*nya sendiri (ii) 32MB SDRAM (iii) 32MB *flash*. Radio yang digunakan adalah TI CC2420 yang berdasarkan IEEE 802.15.4 ZigBee, dimana perangkat dengan standar ini pada PHY dan MAC *layer* beroperasi pada daya rendah dan radio jarak pendek, disasarkan pada *control* dan *monitoring applications*. CC2420 ini juga mendukung *data rate* 250kbps dengan 16 *channel* pada frekuensi 2.4GHz [11].

Sebelumnya Imote2 bekerja pada Intel Platform X. Setelah Imote2 pindah ke Crossbow, Intel Platform X tidak lagi dikembangkan karena Crossbow mengeluarkan sistem operasinya sendiri. Sistem operasi yang kebanyakan dikembangkan oleh komunitas seperti SOS. Namun SOS berhenti dikembangkan pada tahun 2008. Sekarang ini yang masih terlihat adalah TinyOS dan Linux [11].

Sebelum adanya *Embedded* Linux, sistem operasi yang digunakan pada Imote2 adalah TinyOS. Kebanyakan publikasi di web menggunakan TinyOS seperti pada [10], [13], dan [14]. Sekarang dikembangkan *Embedded* Linux karena ditemukan batasan-batasan pada TinyOS seperti *complex-routing* pada suatu topologi JSN. Komunitas *Embedded* Linux melihat sistem operasi *Embedded* Linux pada Imote2 dapat mengatasi keterbatasan tersebut. Namun *Embedded* Linux pada Imote2 masih bersifat baru dan sedang dikembangkan [8].

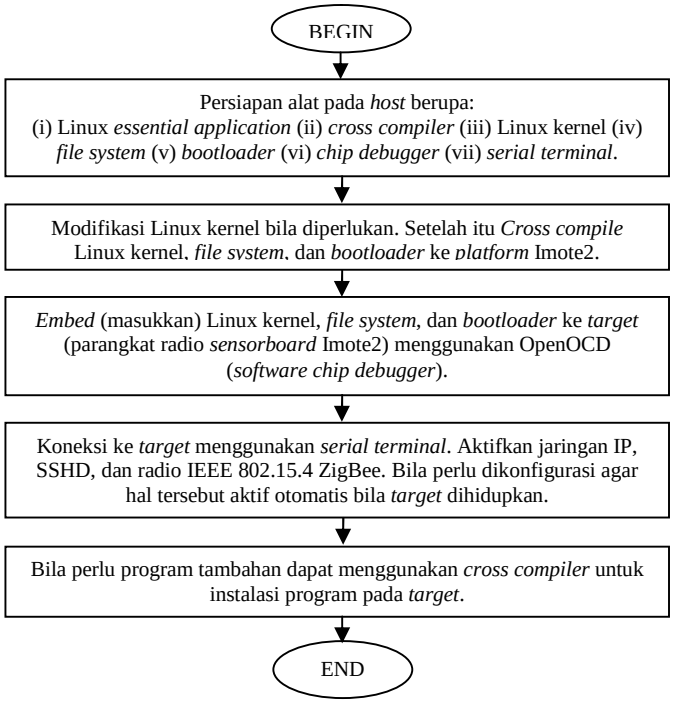
III. IMPLEMENTASI EMBEDDED LINUX

Perangkat yang digunakan sama seperti yang digunakan pada [6-9] dan [13-15], merupakan radio *sensorboard* multimedia *platform* Imote2. Memiliki (i) 256KB SRAM *memory* (ii) 32MB *flash* (iii) 32MB SDRAM (iv) radio terintegrasi dengan 802.15.4 (v) radio *optional* dari SDIO dan UART (vi) 2.4GHz antena (vii) *basic* dan *advanced connectors* seperti 3xUART, I2C, 2xSPI, SDIO, I2S, AC97 *audio*, USB *host*, I/F *camera*, mini USB GPIO. Arsitektur perangkat tersebut dapat dilihat pada Gambar 3.



Gambar. 3 (a) Arsitektur *sensorboard* Imote2 [4] (b) *sensorboard* Imote2 [5]

Pada bagian ini akan dibahas lebih rinci teknis meng*embed* Linux pada *target* yaitu radio *sensorboard* multimedia *platform* Imote2 dengan sumber utama [11]. *Host* adalah Linux OS. Distro Linux dapat digunakan yang mana saja, namun perintah-perintah pada makalah ini berbasis Ubuntu atau Debian. Secara ringkas tahapanya dapat dilihat pada Gambar. 4.



Gambar. 4 Flowchart secara ringkas *embed* Linux pada Imote2

Penelitian [6] dan [7] terutama menyiapkan *cross compiler* dan *chip debugger*. Perintah-perintah yang dimasukkan berdasarkan [13]. Tetapi disini tidak memakai TinyOS. *Cross compiler* yang digunakan berupa *cross compiler* yang telah di*compile* yaitu GCC dan GLIBC dari http://sourceforge.net/projects/imote2-linux/files/imote2-tools-

ISBN: 978-602-7776-72-2

linux. Tetap lebih baik bila meng*compile cross compiler* sendiri agar lebih sesuai dengan kebutuhan perankgat.

Bila *file* terkomproesi dalam bentuk *tape archive gunzip* maka dapat di*extract* dengan perintah berikut:

```
# tar xzvf /lokasi-file/nama-file.tar.gz
```

Bila dalam bentuk *tape archive bunzip2* maka dapat di*extract* dengan perintah berikut:

```
# tar xjvf /lokasi-file/nama-file.tar.bz2
```

Seandainya pada *directory* tersebut tidak diberikan akses penuh maka kondisi *read, write,* dan *execute* perlu diberi angka biner "1". Bila memberi akses penuh maka *rwx* (*read write execute*) adalah "$111_{(2)}$" merupakan "$7_{(10)}$". Maka perintahnya:

```
# chmod –R 777 /lokasi-directory/namadirectory
```

"7" pertama adalah memberi akses *rwx* pada *adminstrator*, yang kedua kepada *user*, dan ketiga untuk *group*, sedangkan "-R" agar berlaku terhadap semua isi pada *directory* tersebut. Penggunaan *cross compiler* ini akan sekaligus dibahas pada saat membentuk Linux kernel. Pada makalah ini langkah-langkahnya adalah sebagai berikut (merupaka satu *directory* dengan *cross compiler* yang telah didownload):

```
#tar xzvf linux-gcc-4.1.2-arm-xscale-linux-gnu-glibc-2.3.3.tgz
#chmod –R 777 arm-xscale-linux-gnu
```

Sebelum memulai sebaiknya men*download dependencies* yang diperlukan:

```
#apt-get install libncurses5-dev libusb-dev libftdi1 libftdi-dev ldconfig mtd-tools ssh
```

Selanjutnya untuk Linux kernel digunakan dari http://www.kernel.org/pub/linux/kernel versi 2.6.29.1 rc 1.1 atau dapat "git" yang tersedia oleh komunitas. Bila diperlukan *extract* dan modifikasi hak akses Linux kernel tersebut. Setelah itu masuk ke *directory* dan atur *cross compiler*. Khusus untuk *compiling Kernel*, pada *file* bernama "Makefile" terdapat baris "ARCH=" dan "CROSS_COMPILE=" yang perlu dideklarasikan (masih kosong). Untuk mendeklarasikan secara *universal* dapat dilihat perintah berikut:

```
# cd /lokasi-directory-Linux-kernel
# export ARCH=arm
# export CROSS_COMPILE=/lokasi-directory-cross-compiler/lokasi-bin/nama-cross-compiler-
```

Pada makalah ini, perintahnya adalah sebagai berikut (Linux *kernel* dan *cross compiler* pada satu *directory*):

```
#tar xzvf linux-2.6.29.1.tar.gz
#chmod –R 777 linux-2.6.29.1
# cd linux-2.6.29.1
# export ARCH=arm
# export CROSS_COMPILE= ../arm-xscale-linux-gnu/bin/arm-xscale-linux-gnu-
```

Dengan perintah tersebut maka ditentukan *platform* adalah "arm" (merupakan *platform* unutuk perangkat-perangkat berukuran kecil) dan menggunana *cross compiler* "arm-xscale-linux-gnu", *directory* "arm-xscale-linux-gnu/bin" terdapat file "arm-xscale-linux-gnu-gcc", "arm-xscale-linux-gnu-g++", dan lain-lain. Untuk mengatur apa saja yang di*compile* dapat dilihat pada *file* "Makefile". Untuk mempermudah digunakan *make menuconfig*. Maka diperlukan "libncurses5-dev". Selanjutnya diperlukan untuk *copy file* "imote2-linux_defconfig" ke *directory* /root menjadi nama ".config".

```
# cp /lokasi-directory-Linux-kernel/lokasi-file-imote2-linux_defconfig/imote2-linux_defconfig /root/.config
```

Pada makalah ini perintahnya (masih pada *directory* Linux kernel):

```
#cp arch/arm/configs/imote2-linux_defconfig /root/.config
```

Untuk *compile kernel*:

```
# make menuconfig
```

Disini dapat diatur apa saja yang ingin di instal.

```
# make jenis-image
```

Jenis image biasanya berupa zImage atau bzImage. Pada makalah ini:

```
# make zImage
```

Selanjutnya membuat *module*:

```
# make module
# make INSTALL_MOD_PATH=$PWD/modules modules_install
```

Perintah ini akan menginstalasi *module* pada *directory* bernama "modules".

Setelah selesai *compile* Linux *kernel* adalah *compile filesystem*. *Source* dari *filesystem* yang digunakan dari http://sourceforge.net/projects/imote2-linux/files/imote2-rootfs. Dibutuhkan juga mkfs.jffs2 *tool* yang terdapat pada mtd-tools. JFFS2 (*Journaled Flash File System* 2) merupakan *file system* yang didesain untuk *flash file* perangkat pada *embedded system*. Setelah *directory file system* di *extract*, *modules* yang telah di*compile* pada Linux-kernel di*copy* pada directory ini.
Untuk membuat 16MB *filesystem*:

```
# mkfs.jffs2 --squash-uid -r ./linux-rootfs -o rootfs.jffs2 -e 0x20000 --pad=0x01000000
```

Untuk membuat 32MB *filesystem*:

```
# mkfs.jffs2 --squash-uid -r ./linux-rootfs -o rootfs.jffs2 -e 0x20000 --pad=0x01DC0000
```

Bahan terakhir yang dibutuhkan setelah Linux *kernel* dan *filesystem* adalah *bootloader*. Pada makalah ini, *bootloader*

yang digunakan tersedia pada http://sourceforge.net/projects/imote2-linux/files/blob-im2.

Bila bahan sudah tersedia maka proses selanjutnya adalah *embed* pada radio *sensorboard platform* Imote2. Seperti pada penelitian [6] dan [7] langkah-langkah untuk *flashing*, pada makalah ini langkah-langkah *flashing* berdasarkan *tutorial* [14] dengan catatan tidak mengikuti langkah-langkah yang menggunakan TinyOS. Untuk *flashing* perlu diinstalasi *driver* JTAG *interface* FTDI. *Package* yang diperlukan pada *host* adalah libusb-dev, libftdi1, libftdi-dev, dan ldconfig. OpenOCD yang digunakan tersedia pada http://downloads.sourceforge.net/project/openocd. Langkah-langkah untuk instalasi OpenOCD sebagai berikut (didalam *directory* OpenOCD yang telah di*extract*):

```
#./configure --enable-ft2232_libftdi
#make
#make install
#chmod –R 777 /lokasi-openocd-yang-telah-diinstalasi
#openocd -f /lokasi-file-configuration -f /lokasi-file-
configurationintelmote
```

Pada makalah ini langkah-langkahnya sebagai berikut:

```
#tar xjvf openocd-0.4.0-rc1.tar.bz2
#cd openocd-0.4.0-rc1
#./configure --enable-ft2232_libftdi
#make
#make install
#chmod –R 777 /usr/local/bin/openocd
```

Hubungkan kabel *USB* seperti pada gambar. (b). Perintah untuk mengkoneksikan *target* dengan *host*:

```
#openocd -f /lokasi/file/configuration –f /lokasi/file/ configuration-
intelmote
```

Pada makalah ini untuk *embed bootloader*, Linux *kernel*, dan *filesystem* seperti berikut (pada *directory* yang berisi *file bootloader*, *directory* Linux *kernel*, dan *directory filesystem*):

```
#openocd –f /usr/local/share/openocd/scripts/interface/
jtagkey.cfg –f board/crossbow_tech_IMote2.cfg
#telnet localhost 4444
>reset halt
>flash protect 0 0 258 off
>flash erase_sector 0 0 258
>flash write_image blob-im2 0x0 bin
>flash write_image linux-2.6.29.1/arch/arm/boot/zImage /zImage
0x00040000 bin
>flash write_image rootfs.jffs2 0x00240000 bin
```

Perintah “reset halt” agar *target* dalam keadaan halted, perintah “flash protect 0 0 258 off” untuk menghilangkan *protect*, perintah “flash erase_sector 0 0 258” untuk menghapus isi pada *sector* tersebut (menghapus isi), dan perintah “flash write source destination” untuk mengisi *target* dari *host*.

Jika semua langkah-langkah tersebut telah dilaksanakan maka Linux telah berhasil di*embed*. Terakhir adalah konfigurasi pada *target* melalui koneksi serial, seperti pada [6], [7], dan [15]. Sebelum lanjut hubungan kabel USB seperti Gambar 5.

Gambar. 5 Cara menghubungkan kabel USB

Applikasi *serial terminal* pada *host* dapat digunakan Putty, GTKterm, dan masih banyak aplikasi lainnya. Untuk koneksi serial dari *host* ke *target* diperlukan pengaturan sebagai berikut:

- *Connection type* : serial
- /dev/ttyUSB1 (atau 0)
- *Speed* 115200
- *Data bit*: 8
- *Stop bit*: 1
- *Parity: none*
- *Flow control*: *XON/OFF*

Alamat IP yang digunakan adalah 192.168.99.101/24. Untuk mengaktifkan alamat IP dan SSHD secara otomatis, dan mengganti alamat IP, langkah-langkahnya sebagai berikut:

```
ln -s /etc/init.d/networking /etc/rc2.d/S10networking
ln -s /etc/init.d/sshd /etc/rc2.d/S11sshd
ln -s /etc/init.d/networking /etc/rc5.d/S10networking
ln -s /etc/init.d/sshd /etc/rc5.d/S11sshd
vi /etc/init.d/networking (ganti IP address)
vi /etc/network/interfaces (ganti IP address)
```

Selanjutnya membuat *script* agar radio hidup secara otomatis. *Script*nya tersimpan pada *directory* “/root/tosmac” dengan nama *loaddriver*, dan isi *loaddriver* sebagai berikut:

```
#!/bin/sh
insmod /lib/modules/2.6.29.1_r1.1/kernel/arch/arm/mach-pxa/ssp.ko
insmod /lib/modules/2.6.29.1_r1.1/kernel/drivers/tosmac/tos_mac.lo
mknod /dev/tosmac c 240 0
```

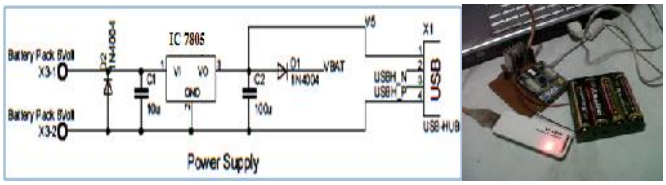
Maka untuk mengaktifkan radio dan dapat mengirim data secara otomatis langkah-langkahnya sebagai berikut:

```
#ln –s /root/tosmac/loaddriver /etc/rc2.d/S12loaddriver
#ln –s /root/tosmac/loaddriver /etc/rc5.d/S12loaddriver
#ln –s /root/tosmac/CntToLeds /etc/rc2.d/S14Transmitter (pengirim data
otomatis)
#ln –s /root/tosmac/CntToLeds /etc/rc5.d/S14Transmitter (pengirim data
otomatis)
```

Tahap *embed* dan konfigurasi telah selesai.

Berbeda dengan penelitian [6] yang menggunakan transmisi radio IEEE 802.15.4 ZigBee, penelitian [7] menggunakan transmisi radio IEEE 802.11 WLAN. Pada [7] dibuatkan *daughter board* yang akan terkoneksi dengan WLAN USB TP-LINK seperti Gambar 6.

ISBN: 978-602-7776-72-2



(a) (b)

Gambar. 6 *Daughter board* [7] (a) skema (b) alat

Pertama perlu diaktifkan USB *host support* dengan menambahkan *script* berikut pada *file* linux-2.6.29.1/arch/arm/mach-pxa/imote2.c. *Scriptnya* sebagai berikut [7]:

```
//baris pertama

#include <mach/ohci.h>

//isi

/*
* Configure USB Host (OHCI)
* For Imote2 the following configuration is used:
* USB Port 1 is used as USB Host
* USB Port 2 is used as USB Gadget (as default for Imote2)
*/
static int imote2_ohci_init(struct device *dev)
{
return 0;
}
static struct pxaohci_platform_data imote2_ohci_platform_data = {
.port_mode = PMM_NPS_MODE,
.init = imote2_ohci_init,
.flags = ENABLE_PORT1 | NO_OC_PROTECTION,
.port_mode = PMM_PERPORT_MODE,
.power_budget = 150, //300
};

//baris terakhir

pxa_set_ohci_info(&imote2_ohci_platform_data);
```

Pada “make menuconfig” dikonfigurasi sebagai berikut [7]:

1. Konfigurasi modul USB-*Host*.
   Device Drivers >USB support >Support for Host-side USB <*>
   >USB device filesystem [*]
   >USB device class-device (DEPRECATED)[*]
   >USB Monitor<*>
   >OHCI HCD support<*>
2. Konfigurasi modul *Wireless* LAN 802.11.
   Networking support >wireless >Improved wireless configuration API (M)
   >nl80211 new netlink interface support [*]
   >Wireless extensions sysfs files [*]
   >Common routines for IEEE802.11 drivers (M)
   >Generic IEEE 802.11 Networking (M)
   >Enable LED triggers [*]
   Device Drivers >Network device Support >Wireless LAN
   >Wireless LAN (IEEE 802.11) [*]
3. Konfigurasi modul driver TP-Link WN-321G (rt73).
   Device Drivers >Network Device Support>Wireless LAN
   >Ralink driver support [M]
   >Ralink rt2501/rt73 (usb0 support) [M]
   >Ralink debug output [*]

Selanjutnya mengulang tahap-tahap dari *compile* Linux *kernel* dan *filesystem* hingga *flashing* dan konfigurasi. Pada penelitian [7] menggunakan TL-WN321G (TP-LINK), driver versi Linux dapat di*download* pada situs resminya. Langkah-langkah instalasinya sebagai berikut:

```
#tar xvf TpLink_TL_WN321G_in_Linux.tar
#cd TpLink_TL_WN321G_in_Linux/Module/
#gedit Makefile
```

a. Menambahkan baris “PLATFORM=EMBEDDED”.
b. Menghilangkan baris “PLATFORM=PC” dan “PLATFORM=CMPC”.
c. Mengatur *link* dari sumber *kernel* yang di*compile*.

```
ifeq ($(PLATFORM),EMBEDDED)
LINUX_SRC=../linux-2.6.29.1
endif

#export ARCH=arm
#export CROSS_COMPILE=../arm-xscale-linux-gnu/bin/arm-xscale-linux-gnu-
#make all
```

Proses *compile* akan menghasilkan sebuah file yang bernama “rt73.ko” yang akan digunakan sebagai *module* untuk *driver* Wireless Lan 802.11. Kemudian *copy* data tersebut pada *target* dengan perintah berikut:

```
#scp rt73.ko root@192.168.99.101:/root rt73.ko
```

Selanjutnya pada *target* dibuat *script* agar radio aktif secara otomatis berdasarkan langkah-langkah dari [7].

```
#ssh –l root 192.168.99.101
#vi /etc/rc2.d/S50StartupScript
```

*Script*nya seperti berikut:

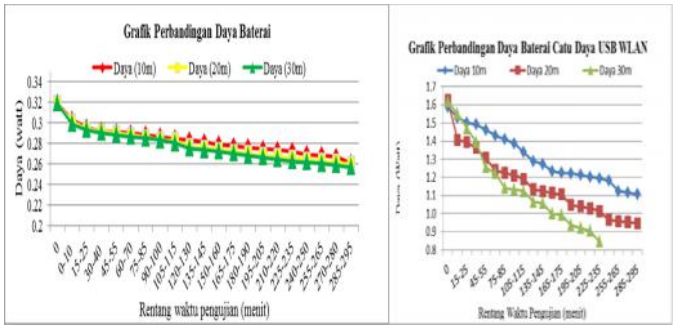
```
#*************************************************************
#*********S50StartupScipt File*******************************
#*********This file configures Wlan on Imote2**************
#install driver
cd /root/
insmod rt73.ko
echo -n 1 > /sys/bus/usb/devices/1-1/bConfigurationValue

#Configure Wlan
sleep 10
ifconfig rausb0 up
iwconfig rausb0 essid imote2
iwconfig rausb0 mode ad-hoc
ip link set rausb0 up
ifconfig rausb0 inet 192.168.1.2
ifconfig rausb0 netmask 255.255.255.0
ifconfig rausb0 gateway 192.168.1.1
#*********EOF S50StartupScript File**************************

#chmod 777 S50StartupScript
```

Tahap konfigurasi bila menggunakan IEEE 802.11 WLAN telah selesai. Jika dibandingkan kedua penelitian tersebut, pada [7] lebih boros baik pada konsumsi *memory* maupun konsumsi daya listrik dibanding [6]. Pada penelitian [7]

memory yang dikonsumsi adalah 16.9MB dari 29.8MB, sedangkan pada [6] hanya dikonsumsi 13.6MB dari 29.8MB. Konsumsi daya listrik dapat dilihat pada Gambar 7.



(a) (b)

Gambar. 7 Konsumsi daya listrik pada *target* (a) perangkat [6], dan (b) perangkat [7]

IV. SIMPULAN

Dari makalah ini dapat ditarik kesimpulan sebagai berikut:

1. Linux *kernel* di*cross-compile* ke *platform* Imote2 terlebih dahulu. Proses selanjutnya membentuk *filesystem* dan menyiapkan *bootloader*.
2. Untuk *embed bootloader*, Linux *kernel*, dan *filesystem* pada *target* melalui *interface* JTAG menggunakan *software* OpenOCD. Prosesnya disebut *flashing*.
3. Konfigurasi *target* meliputi pengaktifkan secara otomatis alamat IP, SSHD, dan radio melalui koneksi serial, dengan membuat *link* ke *script* konfigurasi di RC *level* 2 dan *level* 5.
4. Dari penelitian [6] dan [7] penggunakan media transmisi dengan standar IEEE 802.11 WLAN pada *platform* Imote2 lebih boros *memory* dan daya listrik dibanding menggunakan media transmisi dengan standar IEEE 802.15.4 ZigBee.

UCAPAN TRIMAKASIH

Penelitian ini dibiayai dari dana PNBP Universitas Udayana dengan Surat Perjanjian Pelaksana Penugasan Penelitian NO: 74.118/UN14.2/PNL.01.03.00/2013, tanggal 16 Mei 2013.

REFERENSI

[1] H. Y. Shwe, C. Wang, P. H. J. Chong, A. Kumar. "Robust Cubic-Based 3-D Localization for Wireless Sensor Networks," *wireless sensor network*, vol. 5, no. 9, hal. 169-179, September 2013. [online]. Tersedia: www.scirp.org. [Diakses: 12 Oktober 2013]

[2] http://upload.wikimedia.org/wikipedia/commons/thumb/2/21/WSN.svg/537px-WSN.svg.png. [Diakses: 14 Oktober 2013]

[3] N. P. Sastra. "Wireless Sensor Network," 18 Desember 2009. [Entri Blog]. Blog Wireless Sensor Network Nyoman Putra Sastra. Tersedia: http://staff.unud.ac.id/~putra/2009/12/18/wireless-sensor-network.html. [Diakses: 14 Oktober 2013].

[4] N. P. Sastra, Wirawan, G. Hendrantoro, "Virtual View Image over Wireless Visual Sensor Network," *Telkomnika*, vol.9, no.3, hal. 489-496, Desember 2013. [online]. Tersedia: http://journal.uad.ac.id/index.php/TELKOMNIKA/article/view/1286/677. [Diakses: 14 Oktober 2013].

[5] N. P. Sastra, D. M. Wiharta, I. M. O. Widyantara, Wirawan. "Modeling Wireless Visual Sensor Network with a Low Energy Consumption for Desired Image Quality and View Point," academia.edu shared research [online]. Tersedia: http://www.academia.edu/831948/Modeling_Wireless_Visual_Sensor_Network_with_a_Low_Energy_Consumption_for_Desired_Image_Quality_and_View_Point. [Diakses: 14 Oktober 2013].

[6] I. M. Wiasta, "Performasi Platform Imote2 pada Jaringan Sensor Nirkabel," Laporan Tugas Akhir, Jurusan Teknik Elektro., Universitas Udayana, 2012.

[7] F. S. Natha, "Performasi Platform Imote2 Menggunakan Standar 802.11 pada Jaringan Sensor Nirkabel," Laporan Tugas Akhir, Jurusan Teknik Elektro., Universitas Udayana, 2012.

[8] N. P. Sastra, Wirawan, G. Hendrantoro, "Design and Implementation of Wireless Multimedia Sensor Network Nodes Based on Linux OS," academia.edu shared research [online]. Tersedia: http://www.academia.edu/454554/Design_and_Implementation_of_Wireless_Multimedia_Sensor_Network_Nodes_Based_on_Linux_OS. [Diakses: 14 Oktober 2013].

[9] N. P. Sastra. "Test Capture Image pada Intelmote 2 aka My First IMB400 Image," 18 April 2010. [Entri Blog]. Blog Wireless Sensor Network Nyoman Putra Sastra. Tersedia: http://staff.unud.ac.id/~putra/2010/04/18/test-capture-image-pada-intelmote-2-aka-my-first-imb400-image.html. [Diakses: 18 September 2013].

[10] Stanford, "Imote2," stanford.edu [online]. Tersedia: http://tinyos.stanford.edu/tinyos-wiki/index.php/Imote2 [Terakhir dimodifikasi: 15 Mei 2013, 14:07].

[11] Jorg Kasteleiner, "Principles of applying Embedded Linux on Imote2," Diploma Thesis, Faculty of Computer Science and Engineering., University of Applied Sciences Frankfurt am Main, 2010.

[12] "WSN_Imote2_HW_Bundle_Datasheet," Crossbow Technology Inc, San Jose, California.

[13] N. P. Sastra. "Langkah-Langkah Instalasi TinyOS 2.1.0 Intel mote 2 pada Ubuntu 8.04/9.04/9.10," 18 Desember 2009. [Entri Blog]. Blog Wireless Sensor Network Nyoman Putra Sastra. Tersedia: http://staff.unud.ac.id/~putra/2009/12/18/langkah-langkah-instalasi-tinyos-untuk-intel-mote2.html. [Diakses: 18 September 2013].

[14] N. P. Sastra. "Flashing Program pada Intelmote2," 17 April 2010. [Entri Blog]. Blog Wireless Sensor Network Nyoman Putra Sastra. Tersedia: http://staff.unud.ac.id/~putra/2010/04/17/flashing-program-pada-intelmote2.html. [Diakses: 18 September 2013].

[15] N. P. Sastra. "Tutorial Instalasi Linux embedded system pada intel mote2 (imote2) board," 16 Juni 2010. [Entri Blog]. Blog Wireless Sensor Network Nyoman Putra Sastra. Tersedia: http://staff.unud.ac.id/~putra/2010/06/16/tutorial-instalasi-linux-embedded-system-pada-intel-mote2-imote2-board.html. [Diakses: 18 September 2013].

ISBN: 978-602-7776-72-2 © Universitas Udayana 2013