Kernel Size Reduction For Embedded System

Short Description:

I am proposing this project to reduce the memory foot print of Freebsd kernel. FreeBSD kernel is huge considering embedded system and lots of work need to be done to reduce the kernel size. So I decide to take the responsibility to reduce FreeBSD kernel size.

Project Title: Kernel Size Reduction for Embedded
Name: Amit Rawat
Email: aamitrawat9@gmail.com, aamitr4@gmail.com.
Phone:+919882111750
IM/IRC: *amaraw*
Languages(spoken and written): English

Availability:

I will spend 42 hours per week. My summer vacation is from 10-5-2012 to 26-7-2012 during which I will be completely available .From 26-7-2012 onward my college get open so I have to attend classes (8.00am to 5.00pm) during those days but I will give my Hundred percent.

Bio:

My name is Amit Rawat and I am from the holy city of Haridwar, India which is situated near the bank of river Ganga. I am a 3rd student pursuing an undergraduate degree course with Electronics and Communication Engineering as major from National Institute of Technology, Hamirpur, Himachal Pradesh, India. Apart from having lots of interest in computing in general, I have been a free software enthusiast from past 2 years and creating awareness in and around my college about Free and Open Source Software. I am using FreeBSD for last two year and am highly impressed by its performance and the community support.

I am pretty good in c language and debugging. I am having experience of working with Freebsd code. I read it when I am free and try to understand what function it performs. I have Freebsd stable installed on my system. For testing I use FreeBSD current which I run on Virtualbox.

For completing this project, require good skills which include good understanding of C language, How different portion of kernel code works and How they collaboratively work , Embedded systems and knowledge about their architecture apart from these technical skills patience, commitment and determination to complete this project is also important.

Project Description:

In recent years, embedded systems have become increasingly complex. For example mobile phones which were earlier used for messaging and phone calls now these days use for surfing net, taking pictures, play music and movies, GPRS facility etc. The number of these devices has raised exponentially, so trend of use of general purpose OS is increasing. FreeBSD is freely available, and it is open source gives developers full control, allowing them to adapt the OS. But the only problem is the size because FreeBSD was developed for working on servers and work stations where memory is plenty whereas in case of embedded systems the memory is constrained.

Idea:

a. Reducing the static memory foot print

a.1 Replacing uninitialized static variable

kern_umtx.o (Large text and bss segment).
local_apic.o
atkdb.o (Large bssm and data section).
db_input.o, db_script.o, db_watch.o (large bss segment).
aicasm_macro_scan.o (large bss segment).
aicasm_scan.o (large bss segment).
vfs
kern_lockf
kern_mtxpool
minidump_machdep
syscons
sched_ule
entry (raid)

e.t.c

a.2 Replacing initialized static variable with local variable

kern_jail
kern_linker
kern_malloc
kern_poc
kern_sig

e.t.c

b. Replacing the large inline code with functions
For example in atpic.c

```
 #define ATPIC(io, base, eoi, imenptr)                      \
     { { atpic_enable_source, atpic_disable_source,         \
          (eoi), atpic_enable_intr, atpic_disable_intr,     \
          atpic_source_pending, NULLm_atpic_resume,         \
          atpic_config_intr, atpic_assign cpu}, (io), (base)\
          IDT_IO_INTS+ (base), (imenptr) }
```

c. Reducing the Dynamic Memory foot print for eg. vnode, FFS inode, DIRHASH e.t.c

d. Making a system call center where user can add and remove system calls as per his need.

e. Removing code that remain and compile with kernel even not in use (ata_all.o, ata_da.o, ata_pmp.o, ata_xpt.o, posix alc code e.t.c).

f. Replacing uint64_t, int64_t > uint32_t, int32_t > int16_t, uint16_t > uint8_t, int8_t with lower one if possible.

g. Provide option to for compress file system.

Observable Changes in FreeBSD: When this project will get completed there will a huge reduction in the size of FreeBSD kernel. FreeBSD is going to have a syscall center where user can select and remove the system call he needed. User can use compress file system if there is size constraints.

Deliverables:

a) Providing patches for reducing the static memory foot print and side by side work on system call center.
b) Providing patches for reducing the dynamic memory foot print and side by side work on system call center.
c) Removing unwanted code which always compile with kernel.
d) Implementing (f) part.
e) Finishing the system call center.
f) Investigating compile time flag,write documentation.

Test Plan:

1. I will use virtual box for testing purpose.
2. When my work finish I will test it on real hardware.
3. I will make a script that will store data on file and compare the old kernel result with the new one. So difference can be observed.