ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Porting Embedded Linux to ARM Processor

V.V. Hanuman Prasad, M.Tech
Assistant Professor
Department of ECE
Laqshya Institute of Technology & Sciences,
Tanikella(V), Konijerla (M), Dist:Khammam,
Telangana, India.

P. Sreekanth, M.Tech
Assistant Professor M.Tech
Department of ECE
C.V.R College of Engineering, Vastunagar,
Mangalapalli(V), Ibrahimpatnam(M) R.R.Dist,
Telangana , India.

ABSTRACT

In the realm of embedded technologies ARM (Advanced RISC Machine) is very popular. According to Wikipedia around 70% of all 32 bit embedded CPU's are based on ARM architecture. Its usage is growing in cell phones, PDA's, GPS devices and RFID systems. The Embedded modules, based on ARM, can become very complex machines since these are meant to support varied tasks such as memory management, process management and peripheral interfaces. For seamless integration of these functional modules an OS has to be ported on these ARM based CPUs. For every new CPU architecture, the OS has to be customized, compiled and burnt into the core .With the coming of age of Linux as an embedded OS all this has changed quite significantly. Being in Open Source domain, Linux kernel can be freely downloaded and compiled for any system architecture and this includes ARM based systems also.

"This project describes the details of porting of embedded Linux on ARM core."

PORTING PROCESS

In software engineering, **porting** is the process of adapting software so that an executable program can be created for a computing environment that is different from the one for which it was originally designed (e.g. different CPU, operating system, or third party library). The term is also used when software/hardware is changed to make them usable in different environments.

Software is portable when the cost of porting it to a new platform is less than the cost of writing it from scratch. The lower the cost of porting software, relative to its implementation cost, the more portable it is said to be.

The number of significantly different CPUs and operating systems used on the desktop today is much smaller than in the past. The dominance of the x86 architecture means that most desktop software is never ported to a different CPU. In that same market, the choice of operating systems has effectively been reduced to three: Microsoft Windows, Mac OS/Mac OS X, and Unix/Linux. However, in the embedded systems market, portability remains a significant issue.

BOOTING PROCESS

ANALYSIS OF U-BOOT

There are two stages in the progress of starting up U-Boot

1) Stage 1 uses assemble code, and the following things are completed:

- Initialize hardware device:
- Prepare RAM space for loading stage 2:
- Copy stage 2 to RAM space:
- Initialize heap region:
- Jump to C entrance of stage 2:

2) Stage 2 uses C code to accomplish complex functions,and it has better readability and portability.

The following things are completed:

a)Allocate RAM space for U-Boot:

b) Set SMC controller.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

c)Test the system memory map:

d) Copy kernel image and root file system image from Flash to RAM.

e) Set booting parameters for kernel:

f) Enable interrupt, call for the kernel.

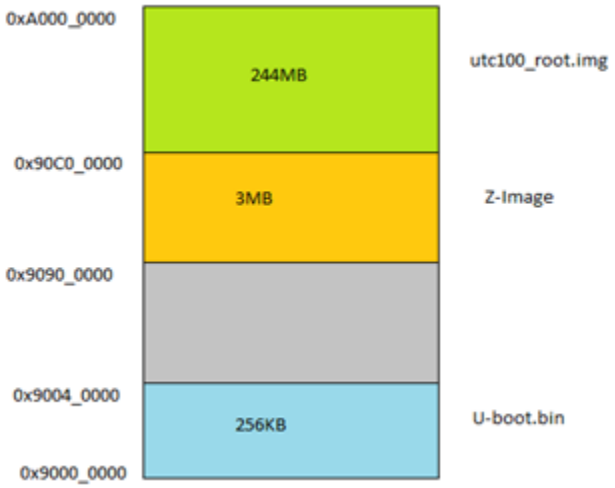


Fig U-Boot memory

FLOW CHART

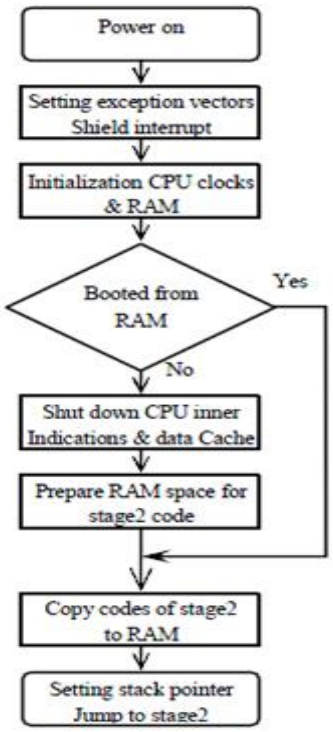


Fig Start Flow chart of Stage1 of u-boot

The first stage of the initialization is to complete equipment such as CPU mode setting for the management mode, turning off the watchdog, setting frequency, close MMU and the CACHE, the initial connection bit wide memory, speed, refresh rate, etc.. As RAM loading space for the second phase of the code, the second phase of code into RAM space, set the stack, jump to the second phase of the C program entry point. The first stage usually does not write code in assembly language, energetic, and high efficiency. This phase involves the code files for the platform and the target associated arch /arm /cpu /arm920t/start.S related board/samsung/smdk2440/lowlevel_init.S.

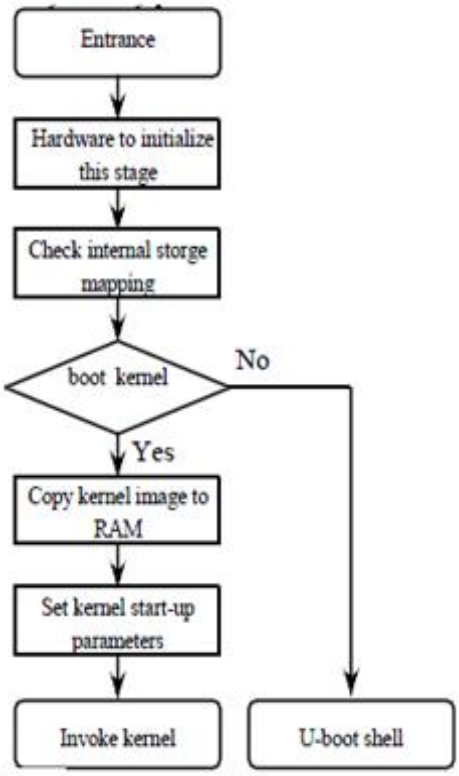
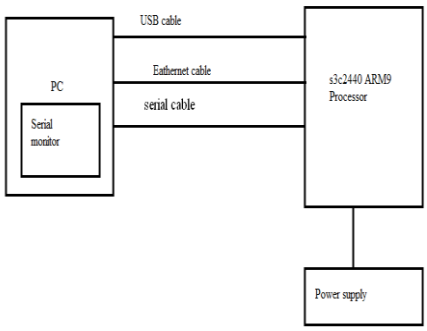


Fig Start Flow chart of Stage 2 of u-boot

Start the second phase of the process diagram shown in Figure 2, the stage is mainly to be used to complete this stage of hardware and memory initialization. If you choose to boot the kernel, then the kernel image and root file system image read from the Flash RAM space, and set the startup environment for the kernel parameters, and then boot the kernel. This phase function is more complicated code, usually written using the C language, so that you can achieve better readability and a higher portability. This phase involves the relevant documents to arch / arm / lib / board.c peripheral target and the corresponding driver files.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

BLOCK DIAGRAM DESCRIPTION



Block diagram of U-boot porting

The MINI2440 Development Board is based on the Samsung S3C2440 microprocessor. Its PCB is 4-layer boarded, equipped with professional equal length wiring which ensures signal integrity.AllMINI2440 boards are manufactured in mass production and released with strict quality control. On startup it directly boots preinstalled Linux by default. There are no extra setup steps or configuring procedures to start the system. It is easy for users to get started. Anyone with very basic knowledge about the C language can become proficient in its development within two weeks. This package also provides detailed documents on how to configure and boot to alternative operating systems. The MINI2440 development board is a 100 x 100(mm) board equipped with a wide variety of connectors, interfaces and ports.

HISTORY AND DEVELOPMENT

- ARM was developed at Acron Computers ltd of Cambridge, England between 1983 and 1985.
- RISC concept was introduced in 1980 at Stanford and Berkley.
- ARM ltd was found in 1990.
- ARM cores are licensed to partners so as to develop and fabricate new microcontrollers around same processor cores.

KEY FEATURES

- ARM processor used here S3C2440.The S3C2440A is developed with ARM920T core, 0.13um CMOS standard cells and a memory complier.
- Its low-power, simple, elegant and fully static design is particularly suitable for cost- and power-sensitive applications.
- It adopts a new bus architecture known as Advanced Micro controller Bus Architecture (AMBA).
- The S3C2440A offers outstanding features with its CPU core, a 16/32-bit ARM920T RISC processor designed by Advanced RISC Machines, Ltd.
- The ARM920T implements MMU, AMBA BUS, and Harvard cache architecture with separate 16KB instruction and 16KB data caches, each with an 8-word line length.
- By providing a complete set of common system peripherals, the S3C2440A minimizes overall system costs and eliminates the need to configure additional components.
- The integrated on-chip functions include 1.2V internal, 1.8V/2.5V/3.3V memory, 3.3V external I/O microprocessor with 16KB I-Cache/16KB D-Cache/MMU.
- External memory controller (SDRAM Control and Chip Select logic),LCD controller (up to 4K color STN and 256K color TFT) with LCD-dedicated DMA.
- 4-ch DMA controllers with external request pins,3-ch UARTs (IrDA1.0, 64-Byte Tx FIFO, and 64-Byte Rx FIFO).
- 2-ch SPls, IIC bus interface (multi-master support),IIS Audio CODEC interface,AC'97 CODEC interface, SD Host interface version 1.0 & MMC Protocol version 2.11 compatible.
- 2-ch USB Host controller / 1-ch USB Device controller (ver 1.1),4-ch PWM timers / 1-ch Internal timer / Watch Dog Timer,8-ch 10-bit ADC and Touch screen interface.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

- RTC with calendar function, Camera interface (Max. 4096 x 4096 pixels input support.

LINUX FEATURES

- Kernel version
- File systems
- Drivers (all open source)
  - Linux applications and utilities
- graphic user interface(open source)
- Qtopia Test Utilities (developed by Friendly ARM, not open source)

SYSTEM SETUP AND CONFIGURATIONS

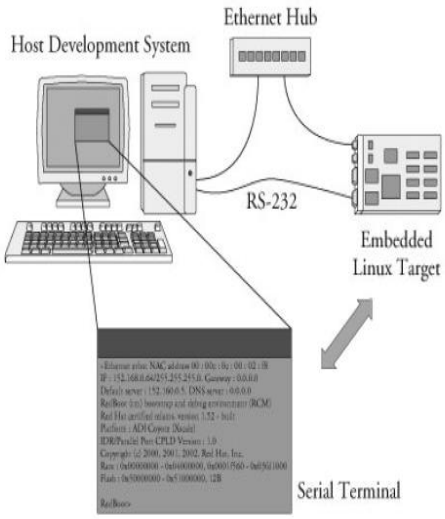
Boot Options: You can select the booting mode by toggling the S2 switch:

When toggling the S2 switch to the "Nor Flash" side the system will boot from on Board nor Flash. When toggling the S2 switch to the "Nand Flash" side the system will boot from on board Nand Flash.

EMBEDDED LINUX

EMBEDDED LINUX SETUP

Often the first question posed by the newcomer to embedded Linux is, just what does one need to begin development? To answer that question, we look at a typical embedded Linux development setup



Embedded Linux Set Up Diagram

Here we show a very common arrangement. We have a host development system, running your favorite desktop Linux distribution, such as Red Hat or SuSE or Debian Linux. Our embedded Linux target board is connected to the development host via an RS-232 serial cable. We plug the target board's Ethernet interface into a local Ethernet hub or switch, to which our development host is also attached via Ethernet. The development host contains your development tools and utilities along with target files normally obtained from an embedded Linux distribution.

LINUX

HISTORY OF UNIX

- UNIX is an Operating System (OS).
- UNIX was developed about 40 years ago i.e., 1969 at AT&T Bell Labs by Ken Thompson and Dennis Ritche.
- It is a Command Line Interpreter.
- It was developed for the Mini-Computers as a time sharing system.
- UNIX was the predecessor of LINUX.

HISTORY OF LINUX

- LINUX was created by Linux Thorvaldsen in 1991.
- LINUX is a open source.
- LINUX is a variant of UNIX.

It's a fast-growing operating system, and it is inexpensive and flexible. Linux is also a major player in the small and mid-sized server field, and it's an increasingly viable platform for workstation and desktop use , as well.

Linux is a clone of the Unix OS that has been popular in academia and many business environments for years.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

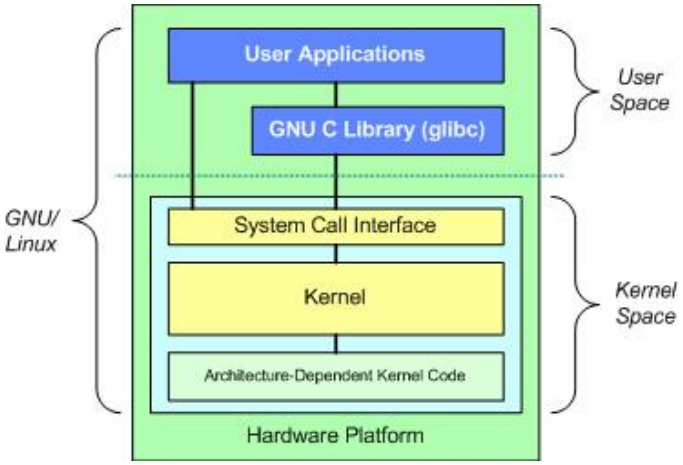


Fig 6.1: Linux operating system

INTRODUCTION TO THE LINUX KERNEL

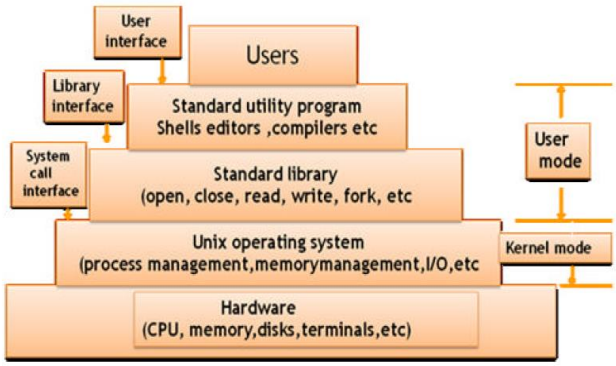


Fig 6.2 Linux kernel

At the top is the user, or application, space. This is where the user applications are executed. Below the user space is the kernel space. Here, the Linux kernel exists.

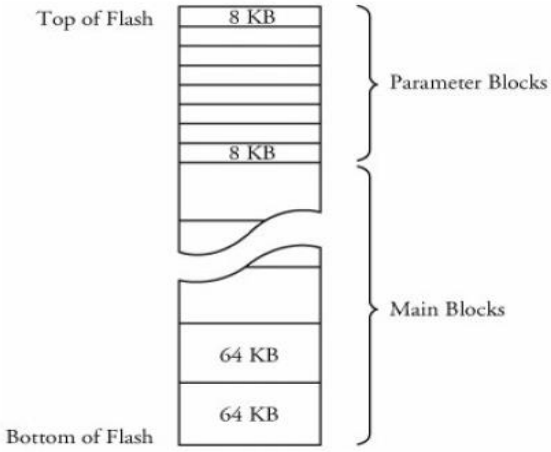
MEMORY CONCEPT

STORAGE CONSIDERATIONS

One of the most challenging aspects of embedded systems is that most embedded systems have limited physical resources. Although the Pentium 4 machine on your desktop might have 180GB of hard drive space, it is not uncommon to find embedded systems with a fraction of that amount. In many cases, the hard drive is typically replaced by smaller and less expensive nonvolatile storage devices. Hard drives are bulky, have rotating parts, are sensitive to physical shock, and require multiple power supply voltages, which makes them unsuitable for many embedded systems.

FLASH MEMORY

Nearly everyone is familiar with Compact Flash modules used in a wide variety of consumer devices, such as digital cameras and PDAs (both great examples of embedded systems). These modules can be thought of as solid-state hard drives, capable of storing many megabytes and even gigabytes of data in a tiny footprint. They contain no moving parts, are relatively rugged, and operate on a single common power supply voltage.



Structure of Flash

To modify data stored in a Flash memory array, the block in which the modified data resides must be completely erased. Even if only 1 byte in a block needs to be changed, the entire block must be erased and rewritten. Flash block sizes are relatively large, compared to traditional hard-drive sector sizes. In comparison, a typical high-performance hard drive has writable sectors of 512 or 1024 bytes. The ramifications of this might be obvious: Write times for updating data in Flash memory can be many times that of a hard drive, due in part to the relatively large quantity of data that must be written back to the Flash for each update. These write cycles can take several seconds, in the worst case. Another limitation of Flash memory that must be considered is Flash memory cell write lifetime. A Flash memory cell has a limited number of write cycles before failure. Although the number of cycles is fairly large (100K cycles typical per block), it is easy to imagine a poorly designed Flash storage algorithm (or even a bug) that can quickly destroy Flash devices. It goes without saying that you should avoid configuring your system

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

COMPILATION PROCESS

COMPILING BOOT LOADER, KERNEL AND ROOT FILE SYSTEM FORSMDK2440

INTRODUCTION TO BOOT LOADER

In embedded system, general firmware like CMOS does not exist. So to boot embedded system for the first time, we have to make boot loader which can adjust well to target board. Boot loader plays a very important part in embedded system. The role of boot loader is explained below.

- Copy kernel to RAM from flash memory, and execute kernel.
- Initialize hardware.
- Boot loader have the function that writing data to flash memory. (Downloading kernel or Ram disk by serial port or other network hardware, data is stored in RAM. But RAM lost all data downloaded if you cut power supply, so to avoid this work you have to store to flash memory.)
- It provides interface to send commands to target board or to inform user's state of target Board.

CONCLUSION:

The project **"title"** been successfully designed and tested**.** Integrating features of all the hardware components used have developed it. Presence of every module has been reasoned out and placed carefully thus contributing to the best working of the unit.

Secondly, using highly advanced IC's and with the help of growing technology the project has been successfully implemented.

Reference:

1] A. El-Sawah, N. Georganas, and E. Petriu, "A prototype for 3-D handtracking and gesture estimation," IEEE Trans. Instrum. Meas., vol. 57,no. 8, pp. 1627–1636, Aug. 2008.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints,"
Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, Nov. 2004

[3] A. Bosch, X. Munoz, and R. Marti, "Which is the best way to organize/ classify images by content?" Image Vis. Comput., vol. 25, no. 6, pp. 778–791, Jun. 2007.

[4] H. Zhou and T. Huang, "Tracking articulated hand motion with Eigen dynamics analysis," in Proc. Int. Conf. Comput. Vis., 2003, vol. 2, pp. 1102–1109.

[5] B. Stenger, "Template based hand pose recognition using multiple cues," in Proc. 7th ACCV, 2006, pp. 551–560.

[6] L. Bretzner, I. Laptev, and T. Lindeberg, "Hand gesture recognition using multiscale color features, hieracrchichal models and particle filtering," in Proc. Int. Conf. Autom. Face Gesture Recog., Washington, DC, May 2002.

[7] A. Argyros and M. Lourakis, "Vision-based interpretation of hand gestures for remote control of a computer mouse," in Proc. Workshop Comput.Human Interact., 2006, pp. 40–51.

Author Details

V.V. Hanuman Prasad, M.Tech
Assist. Professor
Department of ECE
Laqshya Institute of Technology & Sciences,
Tanikella(V), Konijerla (M), Dist:Khammam, Telangana, India.

P. Sreekanth, M.Tech
Assist. Professor M.Tech
Department of ECE
C.V.R College of Engineering, Vastunagar, Mangalapalli(V), Ibrahimpatnam(M) R.R.Dist, Telangana , India.