FUJITSU

shaping tomorrow with you

Running uClinux on ARM Cortex-M3 Platform

Fujitsu Computer Technologies Limited
Sun Wei

June 7th 2012

About Me & My Workplace

- graduated from Univ. of Aizu in Sept. 2011
- entered Fujitsu in Oct. 2011
  - embedded software & hardware
  - server, storage system, network appliance
  - system LSI in cell phone, IC card
- Firmware Engineering Department Ⅰ
- my work
  - kernel building
  - middleware selection & building
  - cross tool chain customization

Contents

- Introduction
  - Background
  - Basics
- Issues
  - Linux Test Project
  - Middleware
  - Cross tool chain
  - Shared library for uClinux/uClibc
  - eXecute In Place
- Conclusion

Contents

- Introduction
  - Background
  - Basics
- Issues
  - Linux Test Project
  - Middleware
  - Cross tool chain
  - Shared library for uClinux/uClibc
  - eXecute In Place
- Conclusion

Background

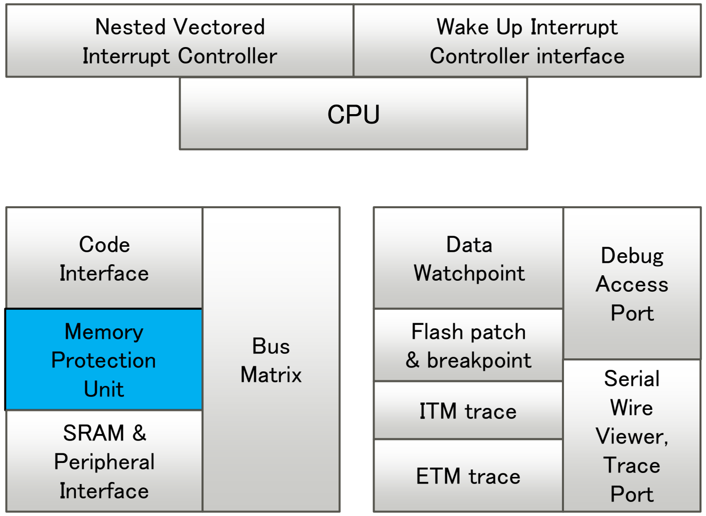
- Applications
  - Merchant Microcontrollers
  - Automotive Control Systems
  - Motor Control Systems
  - White Goods controllers
  - Wireless and Wired Sensor Networks
  - Mass Storage Controllers
  - Printers
  - Network Devices
- Microcontrollers
  - small size
  - low cost
  - multi-function
  - network connection
  - dynamic power

*REF: http://www.arm.com/products/processors/cortex-m/index.php

Background

- ARM Cortex-M Series
  - cost-sensitive solutions for deterministic microcontroller applications
- ARM Cortex-M3 processors
  - designed to deliver industry-leading deterministic behavior, lowest sleep and dynamic power, and smallest area possible whilst maintaining high processing efficiency
  - 1.25 DMIPS/MHz
  - Thumb/Thumb-2
  - Memory Protection Unit

ARM Cortex^{TM} –M3

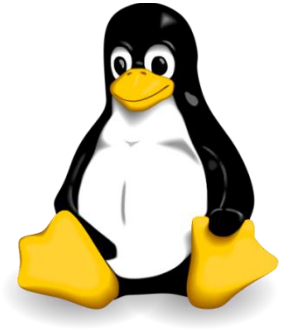


Nested Vectored Interrupt Controller
Wake Up Interrupt Controller interface
CPU
Code Interface
Memory Protection Unit
SRAM & Peripheral Interface
Bus Matrix
Data Watchpoint
Flash patch & breakpoint
ITM trace
ETM trace
Debug Access Port
Serial Wire Viewer, Trace Port

*REF: http://www.arm.com/products/processors/cortex-m/index.php

Background

- uClinux - www.uclinux.org
  - a derivative of Linux 2.0 kernel intended for microcontrollers without MMU
- uClinux-dist distribution
  - uClinux kernel + middleware(including libc)
  - latest version:uClinux-dist-20120401
  - 2.0.39, 2.4.34 and 3.3 kernel sources
  - both the older uC-libc and newer uClibc-0.9.29 libraries



Background

- Microcontrollers based on Cortex-M3
  - Fujitsu FM3
  - NXP LPC1788
  - STmicroelectronics STM32F2
  - Microsemi SmartFusion cSOC
- uClinux on Cortex-M3
  - ARM-Linux, Emcraft, ST
- Problems
  - lack of virtual memory
  - limited resources
- Efforts
- Cooperation

Basics: MMU & MPU

- MMU: Memory Management Unit
  - virtual memory
  - memory protection
- MPU: Memory Protection Unit
  - protection regions
  - overlapping protection regions
  - access permissions
  - exporting memory attributes to the system

Basics: ARM, Thumb & Thumb-2

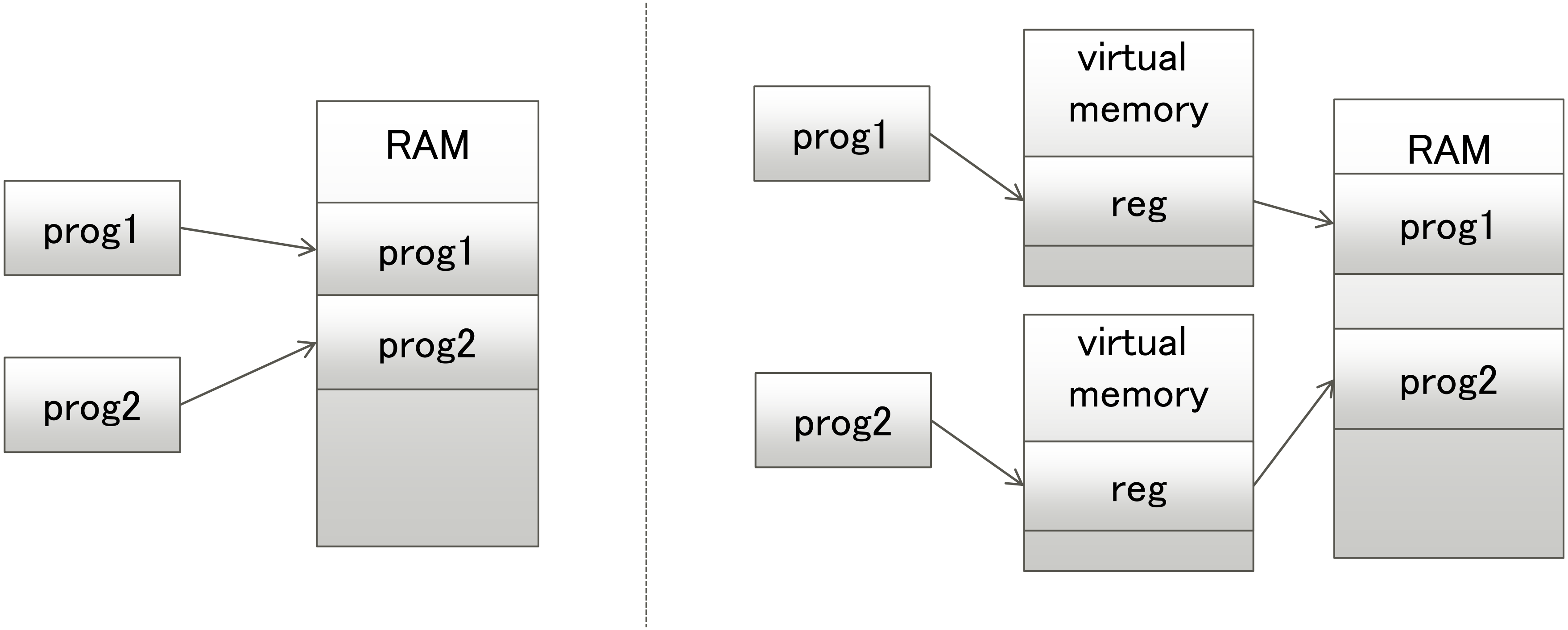
	Bit	Operations	Performance	Code density
ARM	32	rich	better	inferior
Thumb	16	most	inferior	better
Thumb-2	16 and 32	rich	good	good

Thumb-2 is enhancement of Thumb

Cortex-M3 supports Thumb & Thumb-2

Basics: uClinux VS Linux

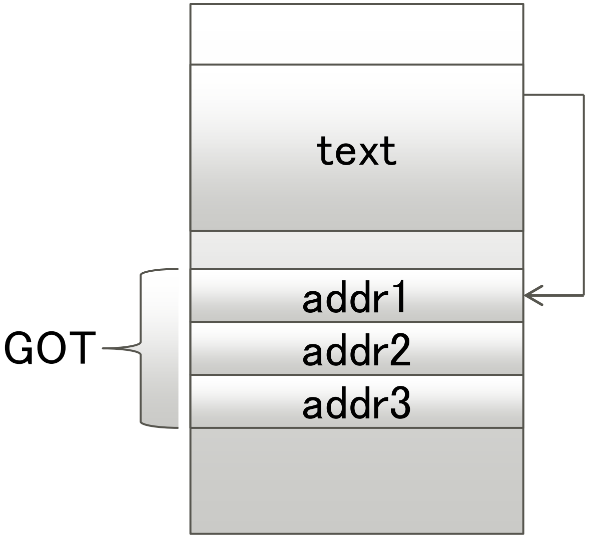
- virtual memory



- uClinux: contiguous and unexpandable
- Linux: discontinuous or swap, expand at runtime

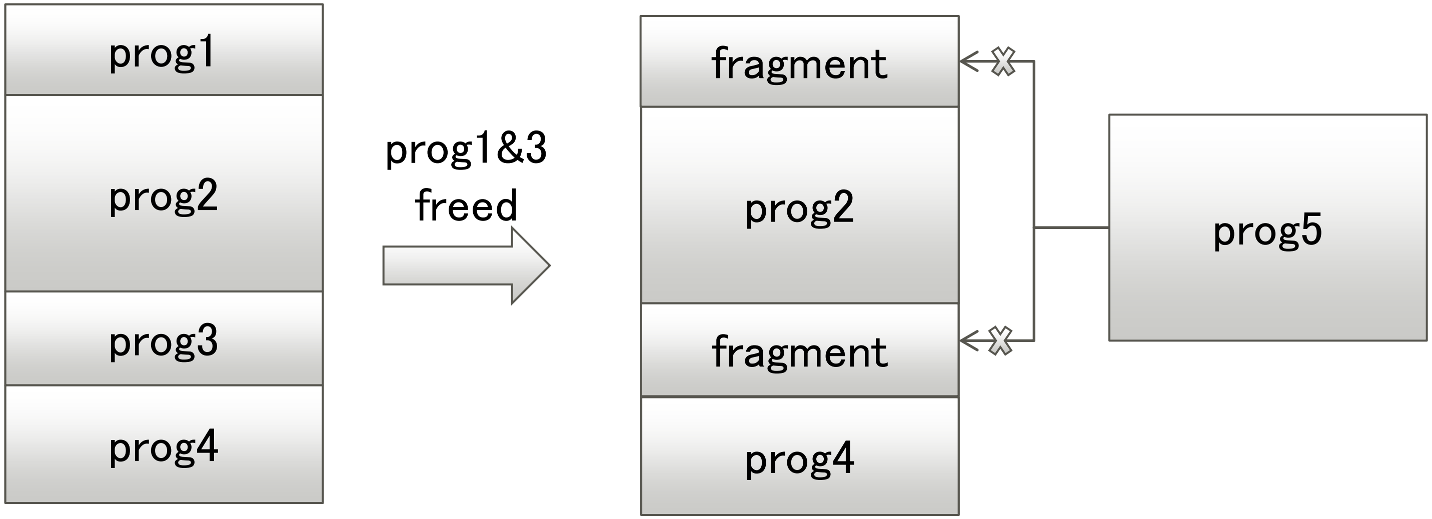
Basics: uClinux VS Linux

- fixed address
  - fix up address references in a program once it is loaded into RAM
- PIC(Position Independent Code)
  - code that uses only relative addressing by a means of GOT, Global Offset Table



Basics: uClinux VS Linux

memory fragment

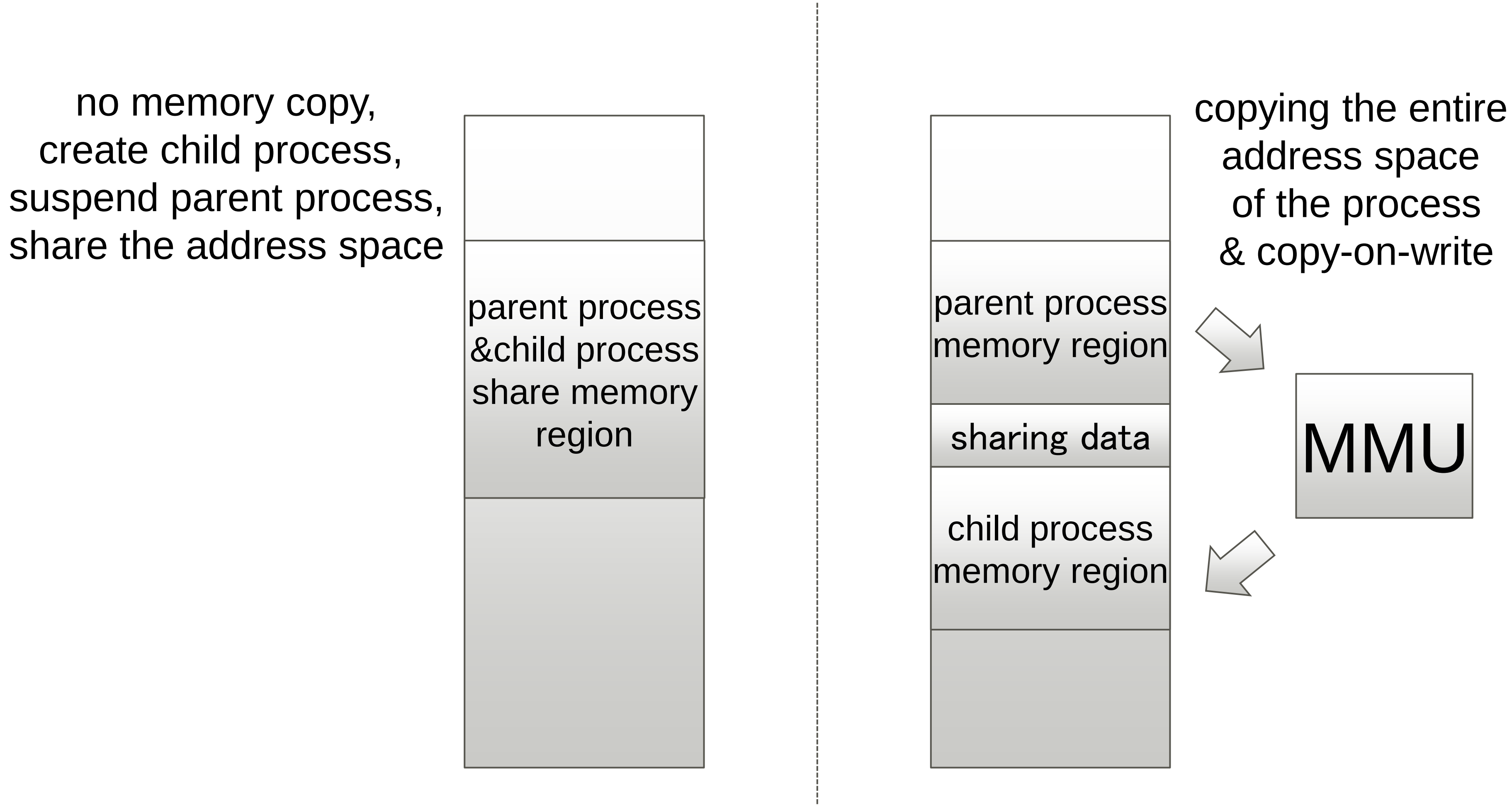


uClinux: preallocated buffer pool

- replacing malloc calls with buffer requests in case of fragment caused by dynamic memory allocation

Basics: uClinux VS Linux

■vfork VS fork


no memory copy,
create child process,
suspend parent process,
share the address space
parent process &child process share memory region
parent process memory region
sharing data
child process memory region
copying the entire address space of the process & copy-on-write
MMU

uClinux cannot use fork due to MMU-less

Basics: uClibc

www.uclibc.org

- a C library for developing embedded Linux systems
  - smaller than glibc/eglibc but supports almost apps
  - licensed under LGPL
  - not supports shared libraries on MMU-less
- Latest version:uClibc-0.9.33.2 May 15 2012

Contents

- Introduction
  - Background
  - Basics
- Issues
  - Linux Test Project
  - Middleware
  - Cross tool chain
  - Shared library for uClinux/uClibc
  - eXecute In Place
- Conclusion

Issues

- problems & efforts
  - Linux Test Project
  - Middleware
  - Cross tool chain
- status
  - Shared library for uClinux/uClibc
  - eXecute In Place

LTP

ltp.sourceforge.net

- Linux Test Project
  - test suites that validate the reliability, robustness, and stability of Linux
  - latest version: Apr. 1st 2012; 3000+ tests
- Concerns about MMU-less
  - FORK_OR_VFORK
  - mmap(): MAP_PRIVATE

(http://mailman.uclinux.org/pipermail/uclinux-dev/2005-March/031528.html)

- our effort
  - 829 test cases: 247 compile NG; 219 test NG
  - compile NG: fork, signal, ustat.h, others
  - test NG: panic/freeze, system calls, etc.
  - results: 574 tested = (351 OK) + (223 NG->OK)

Middleware

- about 330 middleware in uClinux-dist
  - Library: popt, ssl, png…
  - Core application: init, reboot, shutdown…
  - Flash tool: flashw, netflash, recover…
  - File system: mount, fdisk, mke2fs…
  - Network: ftp, http, iptables…
  - Busybox …

Middleware

- not keep up with open source
  - new functionality can not be used
  - vulnerability exists
- Example: dropbear
  - 0.52 onwards has vulnerability CVE-2012-0920
  - 2012/02/22 dropbear-201255 released
  - 2012/04/01 uClinux-dist dropbear 0.43
- our effort
  - busybox1.10.2 -> busybox1.17.0
  - dropbear0.43 -> dropbear0.52 -> dropbear-201255

Cross Tool Chain

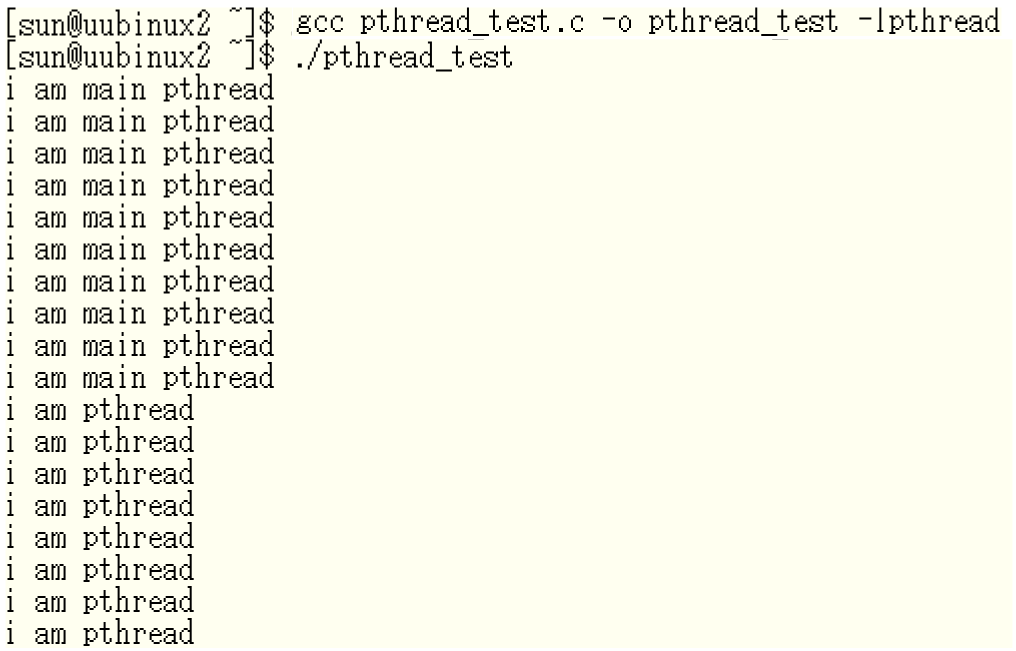
- CodeSourcery
  - arm-2010q1-189-arm-uclinuxeabi
- our effort
  - zlib and gmp
    - required by openssh, openswan
  - libpthread
    - libpthread for ARM not for Thumb
    - uclibc in A2F modified sources of pthread

(http://lists.uclibc.org/pipermail/uclibc/2010-February/043580.html)

Cross Tool Chain

pthread_test

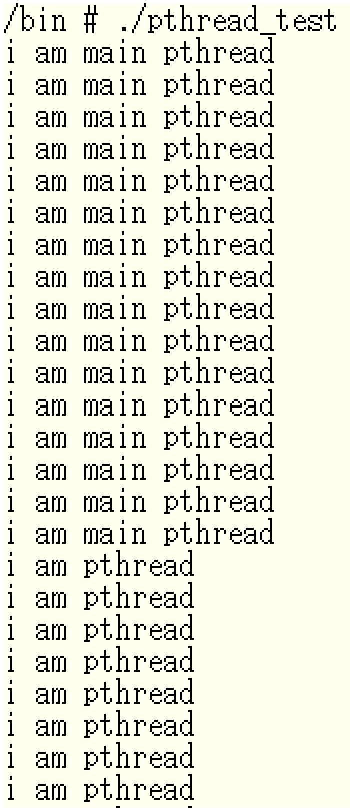
```
#include <stdio.h>
#include <pthread.h>

void* thread_test(void* ptr)
{
    while(1)
        printf("i am pthread¥n");
}

int main()
{
    pthread_t pid;
    pthread_create(&pid, NULL, thread_test, NULL);
    while(1)
        printf("i am main pthread¥n");
    return 0;
}
```



```
[sun@uubinux2 ~]$ gcc pthread_test.c -o pthread_test -lpthread
[sun@uubinux2 ~]$ ./pthread_test
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am pthread
i am pthread
i am pthread
i am pthread
i am pthread
i am pthread
i am pthread
i am pthread
```

Cross Tool Chain

- cross compile
  - arm-uclinuxeabi-gcc pthread_test.c -o pthread_test -mthumb -mcpu=cortex-m3 -Os -march=armv7-m –lpthread
  - mcpu and march
  - link to libpthread for thumb

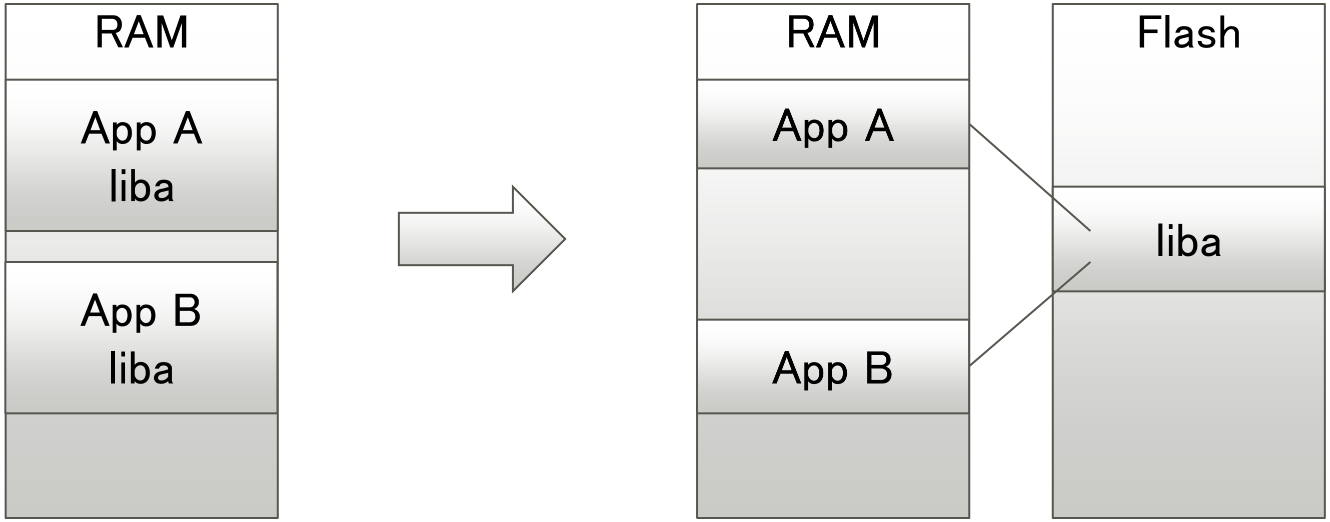


```
/bin # ./pthread_test
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am main pthread
i am pthread
i am pthread
i am pthread
i am pthread
i am pthread
i am pthread
i am pthread
i am pthread
```

Shared library for uClinux/uClibc

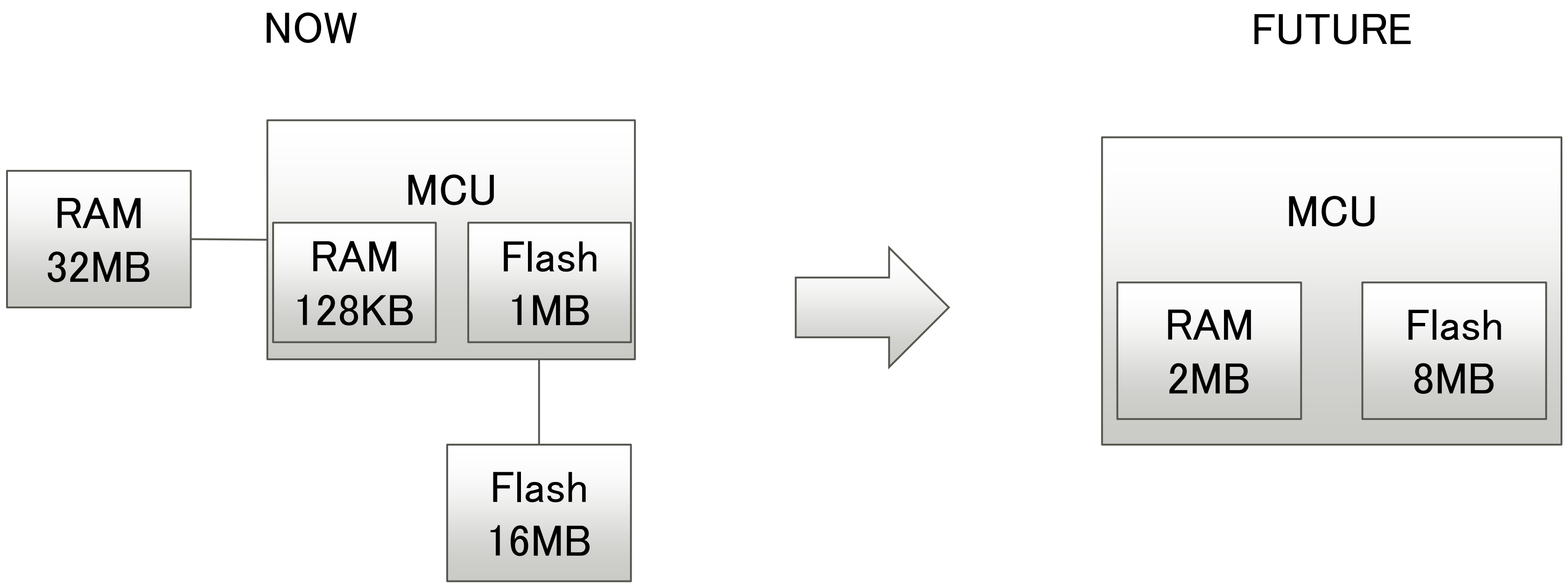
- approaches
  - axLinux contribution
  - RidgeRun: ELF format files
  - SnapGear: uClinux Flat file format

(http://www.linuxfordevices.com/c/a/Linux-For-Devices-Articles/Two-approaches-to-shared-library-support-for-uClinuxuClibc/)



XIP

- eXecute In Place
  - the text segment can reside in flash memory and need not be copied to RAM at all



XIP

- application XIP
  - compiler newer than gcc-4.x cannot be used for XIP uClinux because of the "R_ARM_GOTOFF32" relocation type
  - older compiler not support cortex-m3
- kernel XIP
  - General setup->Kernel Execute-In-Place from ROM =y
  - General setup->Kernel .text physical address = 0xNNNNNNNN

XIP

Emcraft A2F-LNX-EVB

	Start-up time(sec)			ROM size(KB)			RAM size(KB)		
	uboot	kernel	total	kernel	rootfs	total	Available	Reserved	Free
Non-XIP	0.453	1.625	2.078	769.4	200.5	969.9	3,048	1,048	1,324
XIP	0.219	1.984	2.203	773.3	200.5	973.8	3,925	144	1,996
ratio	48.3%	122.1%	106.0%	100.5%	100.0%	100.4%	129.7%	13.7%	150.8%

- faster start-up time of uboot
- no much effect on ROM
- lower RAM reserved by kernel
- more free RAM for applications

Conclusion

- MMU-less architecture for low-cost embedded products
- Many pending issues
  - LTP for uClinux
  - middleware
  - cross tool chain
  - Shared library for uClinux/uClibc
  - application XIP
- Let’s do it!

demonstration

FUJITSU

shaping tomorrow with you