CE5045
Embedded System Design

Embedded Operating System Introduction

https://github.com/tychen-NCU/EMBS-NCU

Instructor: Dr. Chen, Tseng-Yi

Computer Science & Information Engineering

Schema

- ➢ What is an Embedded Operating System?
    - ✓ The Requirements of Embedded O.S.
- ➢ O.S. for Embedded System
    - ✓ Windows Embedded
    - ✓ Embedded Linux
    - ✓ Other Embedded O.S.
- ➢ How to Get Started with Embedded System

Schema

- ➢ What is an Embedded Operating System?
  - ✓ The Requirements of Embedded O.S.
- ➢ O.S. for Embedded System
  - ✓ Windows Embedded
  - ✓ Embedded Linux
  - ✓ Other Embedded O.S.
- ➢ How to Get Started with Embedded System

What is an Embedded O.S.?

- A generic operating system
  - ✓ Process management
  - ✓ Memory management
  - ✓ I/O device management
  - ✓ Network management
  - ✓ Secondary storage management
  - ✓ Security

Characteristics of Embedded O.S.

- An embedded operating system usually has…
  - ✓ Real-time operation
  - ✓ Reactive operation (event driven)
  - ✓ Configurability
  - ✓ I/O device flexibility
  - ✓ Streamlined protection mechanisms
  - ✓ Direct use of interrupts

fedora VxWorks μClinux Windows 10 Pro

ubuntu freeRTOS Windows IoT Real-Time LINUX

Characteristics of Embedded O.S.

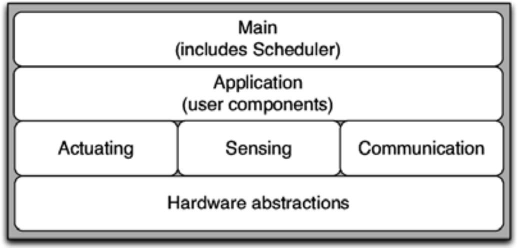
- An embedded operating system usually has…
  - ✓ Real-time operation
  - ✓ Reactive operation (event driven)
  - ✓ Configurability
  - ✓ I/O device flexibility
  - ✓ Streamlined protection mechanisms
  - ✓ Direct use of interrupts

fedora
VxWorks
μClinux
Windows 10 Pro
ubuntu
free RTOS
Windows IoT
Real-Time LINUX

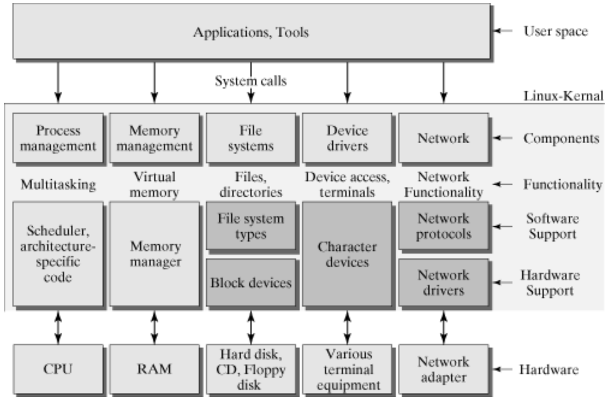
Embedded Operating System is

- ➢ Definition of Embedded O.S.
  - ✓ A simple operating system designed for embedded systems

Tiny O.S. architecture



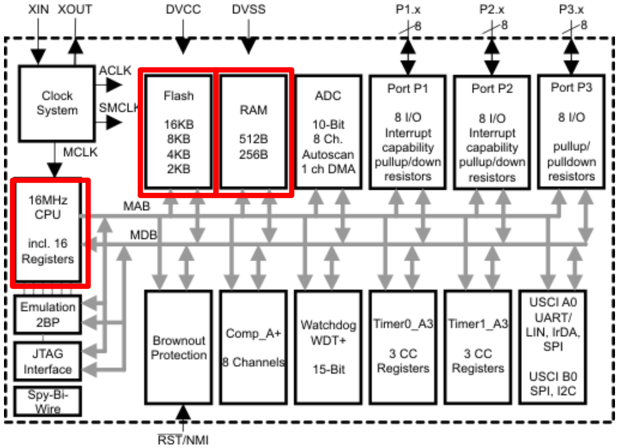
GP Linux architecture



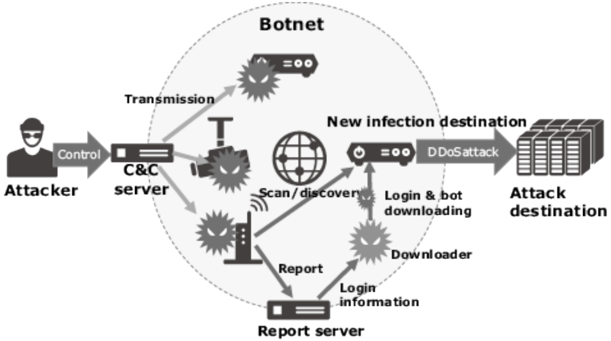
Embedded Operating System is

- Definition of Embedded O.S.
  - ✓ A simple operating system designed for embedded systems
  - ✓ Resource efficiency and reliability

TI MSP430 Specification



Embedded System Security



Embedded Operating System is

- Definition of Embedded O.S.
  - ✓ A simple operating system designed for embedded systems
  - ✓ Resource efficiency and reliability
  - ✓ Time constraint

Hard real-time embedded system

Soft real-time embedded system

Embedded Operating System is

- Definition of Embedded O.S.
  - ✓ A simple operating system designed for embedded systems
  - ✓ Resource efficiency and reliability
  - ✓ Time constraint

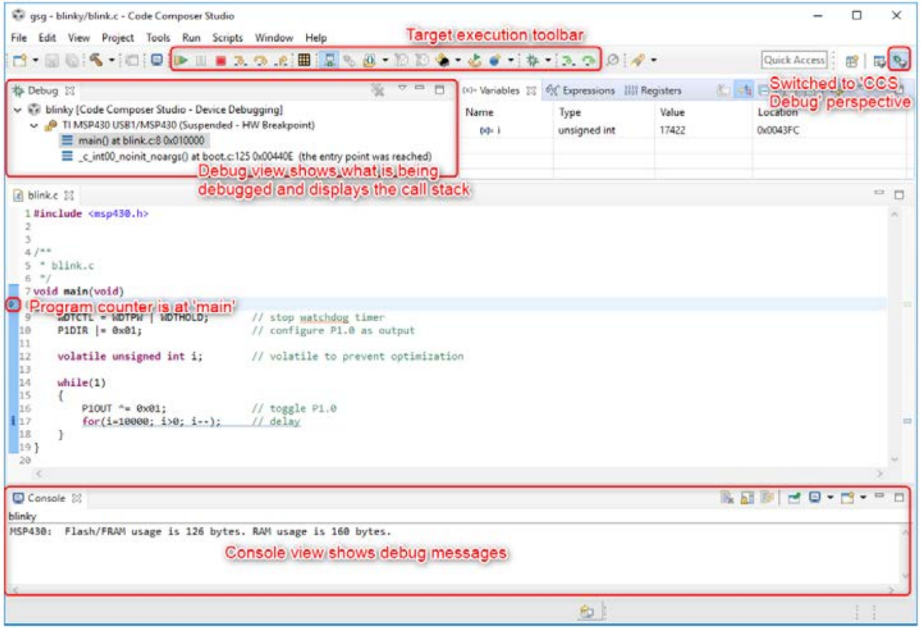
Hard real-time embedded system

Soft real-time embedded system

Difference: Penalty

Embedded Operating System is

- Definition of Embedded O.S.
  - ✓ A simple operating system designed for embedded systems
  - ✓ Resource efficiency and reliability
  - ✓ Time constraint
  - ✓ Generally written in the C language



gsg - blinky/blink.c - Code Composer Studio
File Edit View Project Tools Run Scripts Window Help
Target execution toolbar
Quick Access
Switched to 'CCS Debug' perspective
Debug
Variables
Expressions
Registers
Debug view shows what is being debugged and displays the call stack
blink.c
Program counter is at 'main'
Console
Console view shows debug messages

blinky [Code Composer Studio - Device Debugging]
TI MSP430 USB1/MSP430 (Suspended - HW Breakpoint)
main() at blink.c:8 0x010000
_c_int00_noinit_noargs() at boot.c:125 0x00440E (the entry point was reached)

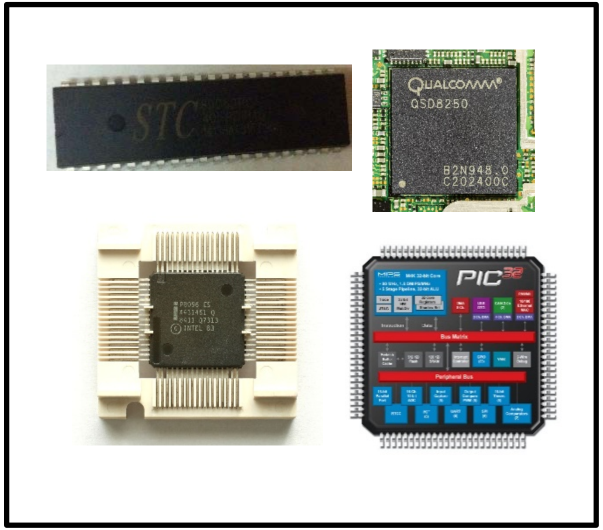
Name	Type	Value	Location
i	unsigned int	17422	0x0043FC

```
#include <msp430.h>


/**
 * blink.c
 */
void main(void)

    WDTCTL = WDTPW | WDTHOLD;           // stop watchdog timer
    P1DIR |= 0x01;                      // configure P1.0 as output

    volatile unsigned int i;            // volatile to prevent optimization

    while(1)
    {
        P1OUT ^= 0x01;                  // toggle P1.0
        for(i=10000; i>0; i--);         // delay
    }
}
```

```
blinky
MSP430:  Flash/FRAM usage is 126 bytes. RAM usage is 160 bytes.
```

Types of Embedded System

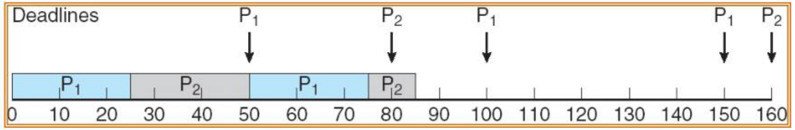
- Embedded systems can be classified based on
  - ✓ Performance and functional requirement
  - ✓ Performance of microcontroller



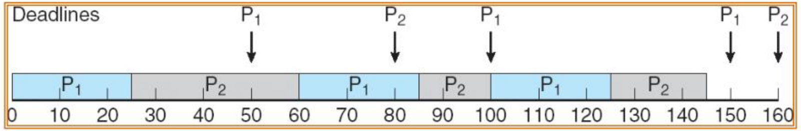
Types of Embedded System

- Embedded systems can be classified based on
  - ✓ Performance and functional requirement
    - **<u>Real-time embedded systems: Hard real-time and soft real-time</u>**
    - Stand alone embedded systems
    - Networked embedded systems
    - Mobile embedded systems
  - ✓ Performance of microcontroller

Rate-monotonic scheduling: misses deadlines



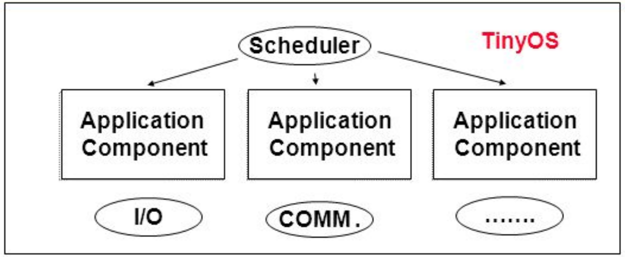
Earliest-Deadline-First scheduling:



$T = \{S, D, P, E\}$
$T_1 = \{0, 50, 50, 25\}$
$T_2 = \{0, 80, 80, 35\}$

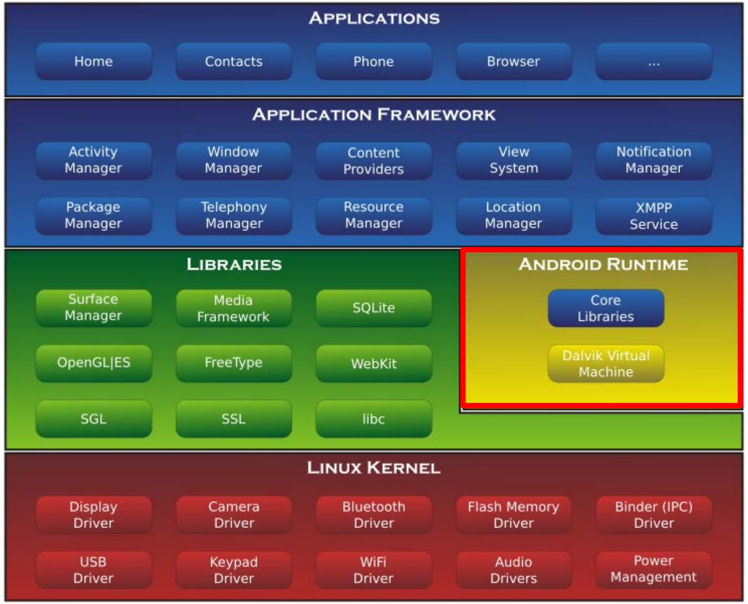
Types of Embedded System

- Embedded systems can be classified based on
  - ✓ Performance and functional requirement
    - Real-time embedded systems: Hard real-time and soft real-time
    - Stand alone embedded systems
    - **Networked embedded systems**
    - Mobile embedded systems
  - ✓ Performance of microcontroller



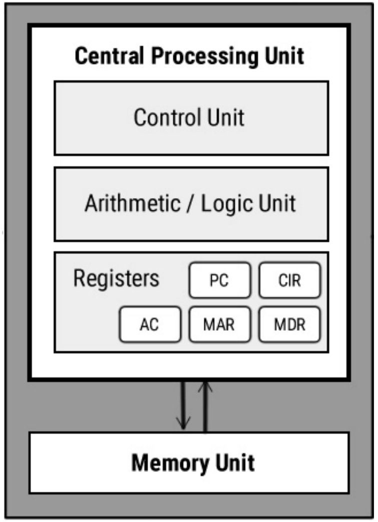
Types of Embedded System

- Embedded systems can be classified based on
  - ✓ Performance and functional requirement
    - Real-time embedded systems: Hard real-time and soft real-time
    - Stand alone embedded systems
    - Networked embedded systems
    - <u>**Mobile embedded systems**</u>
  - ✓ Performance of microcontroller



Types of Embedded System

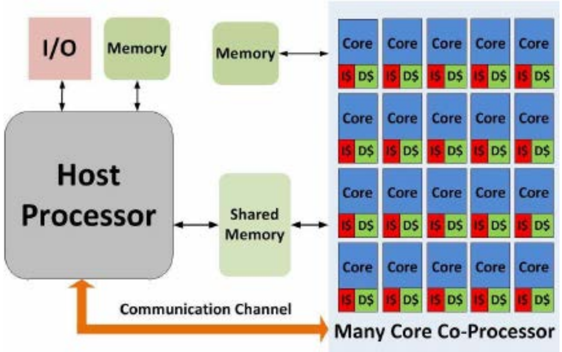
- ➢ Embedded systems can be classified based on
  - ✓ Performance and functional requirement
    - Real-time embedded systems: Hard real-time and soft real-time
    - Stand alone embedded systems
    - Networked embedded systems
    - Mobile embedded systems
  - ✓ Performance of microcontroller
    - 8-bit controller (a single chip microcontroller): Intel 8051
    - 16-bit controller: Intel 8096 and PIC 24
    - 32-bit controller: ARM and PIC 32
    - Heterogeneous SoC



Von Neumann

Types of Embedded System

- Embedded systems can be classified based on
  - ✓ Performance and functional requirement
    - Real-time embedded systems: Hard real-time and soft real-time
    - Stand alone embedded systems
    - Networked embedded systems
    - Mobile embedded systems
  - ✓ Performance of microcontroller
    - 8-bit controller (a single chip microcontroller): Intel 8051
    - 16-bit controller: Intel 8096 and PIC 24
    - 32-bit controller: ARM and PIC 32
    - **Heterogeneous SoC**



Schema

- What is an Embedded Operating System?
    - ✓ The Requirements of Embedded O.S.
- O.S. for Embedded System
    - ✓ Windows Embedded
    - ✓ Embedded Linux
    - ✓ Other Embedded O.S.
- How to Get Started with Embedded System

Developing an Embedded O.S.

- How to build an embedded operating system?
  - ✓ Take an existing O.S. and adapt it for the embedded application.
  - ✓ Design and implement an O.S. intended solely for embedded use

Adapting an Existing O.S.

➢ An existing commercial OS can be used for an embedded system by adding:

- ✓ Real time capability
- ✓ Streamlining operation
- ✓ Adding necessary functionality

Advantage:

- ● Familiar interface

Disadvantage:

- ● Not optimized for real-time and embedded applications

Embedded Windows

- ➢ Common version – Windows 10
  - ✓ Easy of use: Friendly graphic user interface
  - ✓ Available software
  - ✓ Support for new hardware
  - ✓ Plug & Play

Embedded Windows

- ➢ Common version – Windows 10
  - ✓ Easy of use: Friendly graphic user interface
  - ✓ Available software
  - ✓ Support for new hardware
  - ✓ Plug & Play

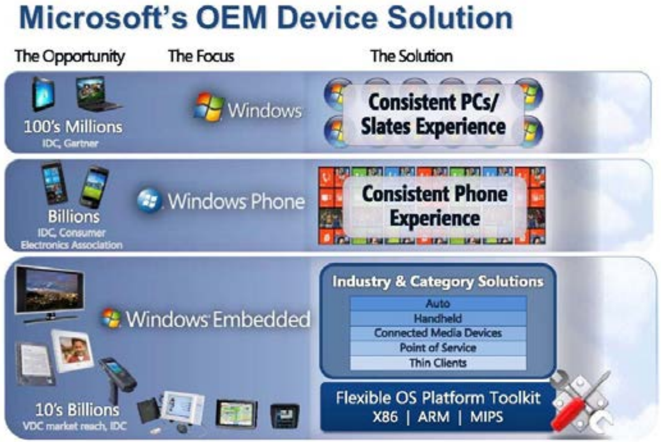
- ➢ But for embedded system?
  - ✓ High resource requirements

Embedded Windows

- Long time ago…
  - ✓ DOS
    - The most famous one: MS-DOS by created by Tim Paterson
    - 16 bits O.S.
    - Single user and single task
    - Application: LED billboard and industry control

Embedded Windows

- Embedded Windows Family
  - ✓ Products:
    - Windows Embedded Standard, Windows Embedded Compact, Windows Embedded Enterprise, Windows Embedded POSReady
  - ✓ Support CPU: x86, ARM, MIPS, …etc
  - ✓ Development tool: Visual studio and windows IoT emulator [link]



What’s Linux

- ➢ Definition of Linux
  - ✓ Linux is the kernel developed and maintained by Linus Torvalds
  - ✓ Based on the Linux kernel, there are many different distributions
  - ✓ Linux Kernel includes
    - ● Controls all hardware
    - ● Provides core system facilities
    - ● Manages system through its lifecycle (next reboot)
    - ● Provides higherlevel abstractions to software

What’s Embedded Linux

What’s Embedded Linux

- Embedded Linux doesn't exist

What’s Embedded Linux

- Embedded Linux doesn't exist
  - ✓ There is no specific kernel for embedded systems
  - ✓ There are, nevertheless, customized kernels specially configured / customized for specific embedded hardware configurations.

What’s Embedded Linux

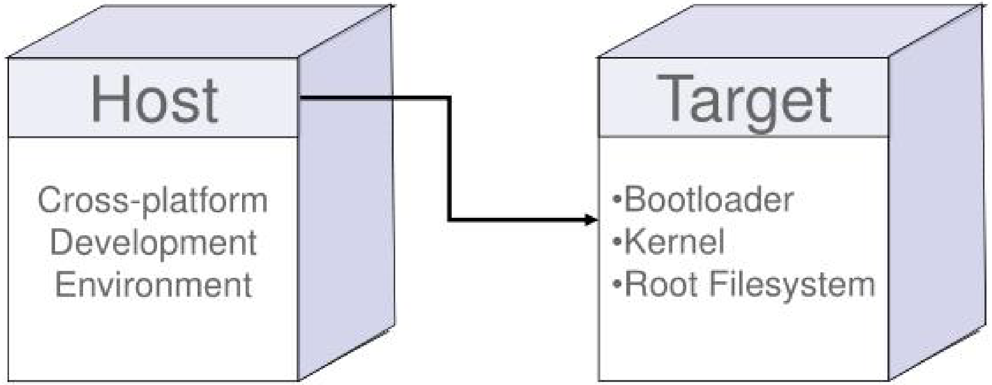
- Embedded Linux doesn't exist
  - ✓ There is no specific kernel for embedded systems
  - ✓ There are, nevertheless, customized kernels specially configured / customized for specific embedded hardware configurations.
- What does exist:
  - ✓ Embedded Linux system

What’s Embedded Linux

- Embedded Linux doesn't exist
  - There is no specific kernel for embedded systems
  - There are, nevertheless, customized kernels specially configured / customized for specific embedded hardware configurations.
- What does exist:
  - Embedded Linux system
    - An embedded system running the Linux kernel
    - Userspace tools & configuration likely to be very different from desktop (uClibc instead of glibc, BusyBox instead of coreutils, etc.)

Cross Compiler

- ➢ A key differentiator between desktop/server and embedded Linux distributions is that desktop and server software is typically compiled on the platform where it will execute
- ➢ Embedded Linux distributions are usually compiled on one platform but are intended to be executed on another
  - ✓ The software used for this purpose is referred to as a cross-compiler

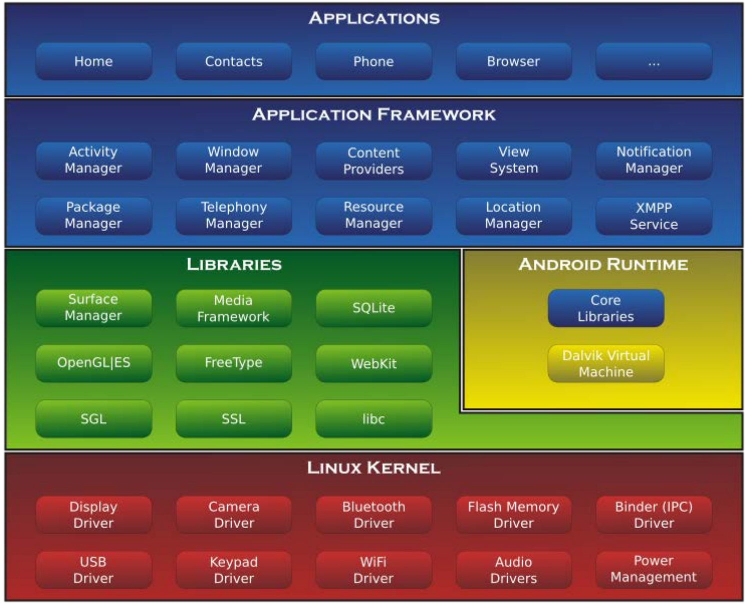


Advantages of Embedded Linux System

- The strengths of an embedded Linux system include
  - ✓ Vendor independence
  - ✓ Varied hardware support
  - ✓ Low cost
  - ✓ Open source

Based on Linux Kernel

- ➢ Android mobile system
  - ✓ Focus of Android lies in the vertical integration of the Linux kernel and the Android user-space components.

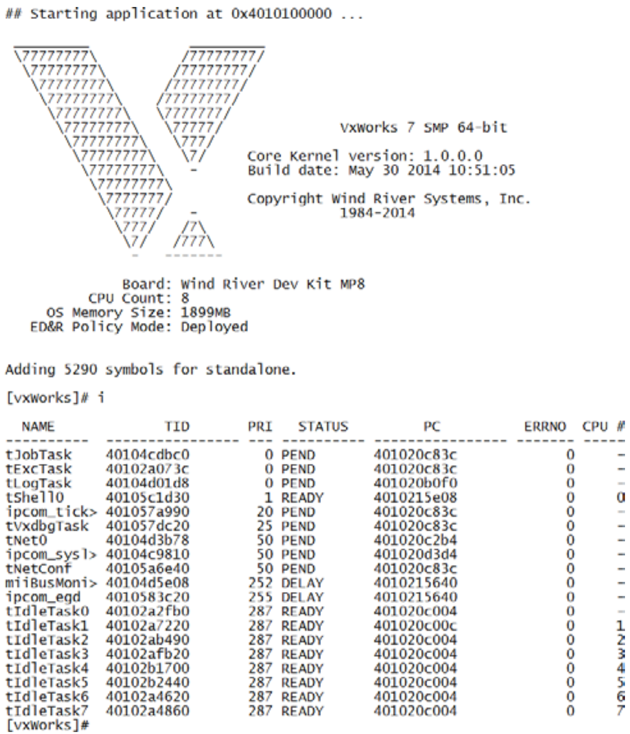


Purpose-Built Embedded O.S.

- Typical characteristics include:
  - ✓ Fast and lightweight process or thread switch
  - ✓ Scheduling policy is real time and dispatcher module is part of scheduler
  - ✓ Small size
  - ✓ Responds to external interrupts quickly
  - ✓ Provides fixed or variable-sized partitions for memory management
  - ✓ Provides special sequential files that can accumulate data at a fast rate

Other Embedded O.S.

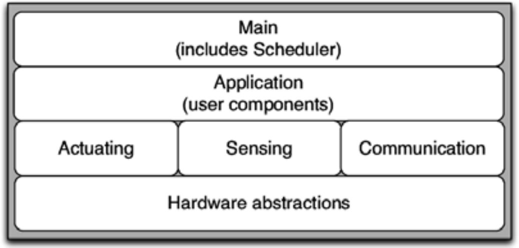
- VxWorks
    - ✓ Created by WindRiver
    - ✓ Support CPU: x86, i960, MIPS, PowerPC
    - ✓ Popularly use in embedded systems
    - ✓ GNU compiler and debugger
    - ✓ Hard real-time



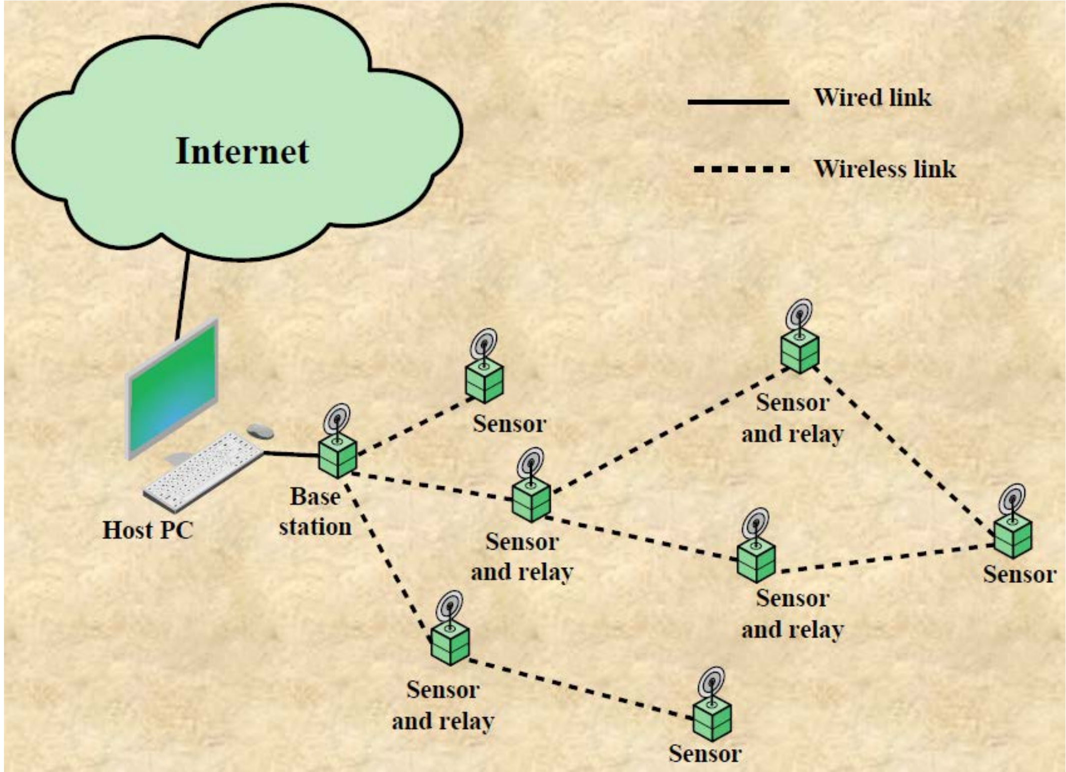
```
## Starting application at 0x4010100000 ...

 _________          _________
 \77777777\        /77777777/
  \77777777\      /77777777/
   \77777777\    /77777777/
    \77777777\  /77777777/
     \77777777\ \7777777/
      \77777777\ \77777/              VxWorks 7 SMP 64-bit
       \77777777\ \777/
        \77777777\ \7/           Core Kernel version: 1.0.0.0
         \77777777\ -            Build date: May 30 2014 10:51:05
          \77777777\
           \7777777/             Copyright Wind River Systems, Inc.
            \77777/ -                       1984-2014
             \777/ /7\
              \7/ /777\
               -  -------

               Board: Wind River Dev Kit MP8
           CPU Count: 8
      OS Memory Size: 1899MB
    ED&R Policy Mode: Deployed

Adding 5290 symbols for standalone.

[vxWorks]# i

  NAME             TID          PRI   STATUS          PC             ERRNO  CPU #
----------  ---------------- --- ---------- ---------------- ------- -----
tJobTask     40104cdbc0         0 PEND       401020c83c             0     -
tExcTask     40102a073c         0 PEND       401020c83c             0     -
tLogTask     40104d01d8         0 PEND       401020b0f0             0     -
tShell0      40105c1d30         1 READY      4010215e08             0     0
ipcom_tick> 401057a990        20 PEND       401020c83c             0     -
tVxdbgTask   401057dc20        25 PEND       401020c83c             0     -
tNet0        40104d3b78        50 PEND       401020c2b4             0     -
ipcom_sysl> 40104c9810        50 PEND       401020d3d4             0     -
tNetConf     40105a6e40        50 PEND       401020c83c             0     -
miiBusMoni> 40104d5e08       252 DELAY      4010215640             0     -
ipcom_egd    4010583c20       255 DELAY      4010215640             0     -
tIdleTask0   40102a2fb0       287 READY      401020c004             0     -
tIdleTask1   40102a7220       287 READY      401020c00c             0     1
tIdleTask2   40102ab490       287 READY      401020c004             0     2
tIdleTask3   40102afb20       287 READY      401020c004             0     3
tIdleTask4   40102b1700       287 READY      401020c004             0     4
tIdleTask5   40102b2440       287 READY      401020c004             0     5
tIdleTask6   40102a4620       287 READY      401020c004             0     6
tIdleTask7   40102a4860       287 READY      401020c004             0     7
[vxWorks]#
```

Other Embedded O.S.

- Tiny O.S.
  - ✓ Streamlines to a very minimal OS for embedded systems
  - ✓ Core OS requires 400 bytes of code and data memory combined
  - ✓ Not a real-time OS
  - ✓ There is no kernel
  - ✓ There are no processes
  - ✓ OS doesn’t have a memory allocation system
  - ✓ Interrupt and exception handling is dependent on the peripheral
  - ✓ It is completely nonblocking, so there are few explicit synchronization primitives
  - ✓ Has become a popular approach to implementing wireless sensor network software

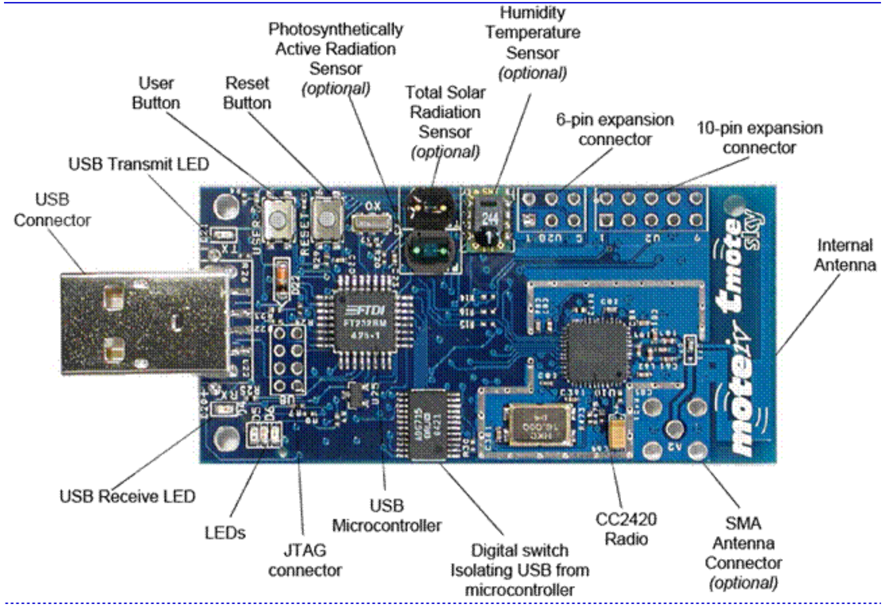


Networked Embedded System



Embedded Platform for TinyO.S.

- Tmote sky
  - ✓ 8MHz Texas Instruments MSP430 microcontroller
  - ✓ 10k RAM, 48k Flash
  - ✓ 250kbps 2.4GHz IEEE 802.15.4



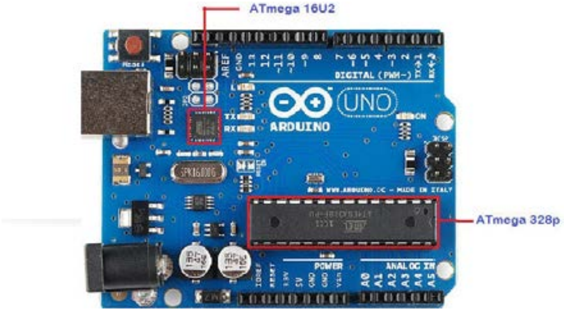
Photosynthetically Active Radiation Sensor (optional)
Humidity Temperature Sensor (optional)
User Button
Reset Button
Total Solar Radiation Sensor (optional)
6-pin expansion connector
10-pin expansion connector
USB Transmit LED
USB Connector
Internal Antenna
USB Receive LED
LEDs
JTAG connector
USB Microcontroller
Digital switch Isolating USB from microcontroller
CC2420 Radio
SMA Antenna Connector (optional)

Schema

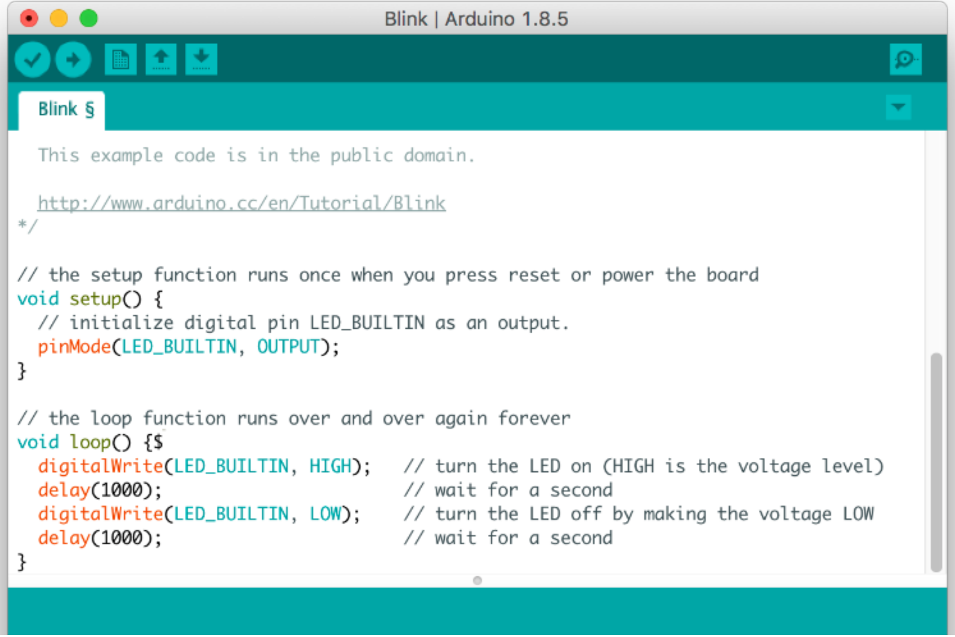
- ➢ What is an Embedded Operating System?
  - ✓ The Requirements of Embedded O.S.
- ➢ O.S. for Embedded System
  - ✓ Windows Embedded
  - ✓ Embedded Linux
  - ✓ Other Embedded O.S.
- ➢ How to Get Started with Embedded System

Arduino Platform

- Arduino R3 UNO
  - Microcontroller : ATmega328p (20 MIPS at 20MHz)
  - Operating Voltage : 5V
  - Input Voltage (recommended) : 7-12V
  - Input Voltage (limits) : 6-20V
  - Digital I/O Pins : 14
  - Analog Input Pins : 6
  - Flash Memory : 32 KB (ATmega328) of which 0.5 KB used by bootloader SRAM 2 KB (ATmega328)
  - EEPROM : 1 KB (ATmega328)
  - Development tool: https://www.arduino.cc/en/main/software



Arduino Development Env.

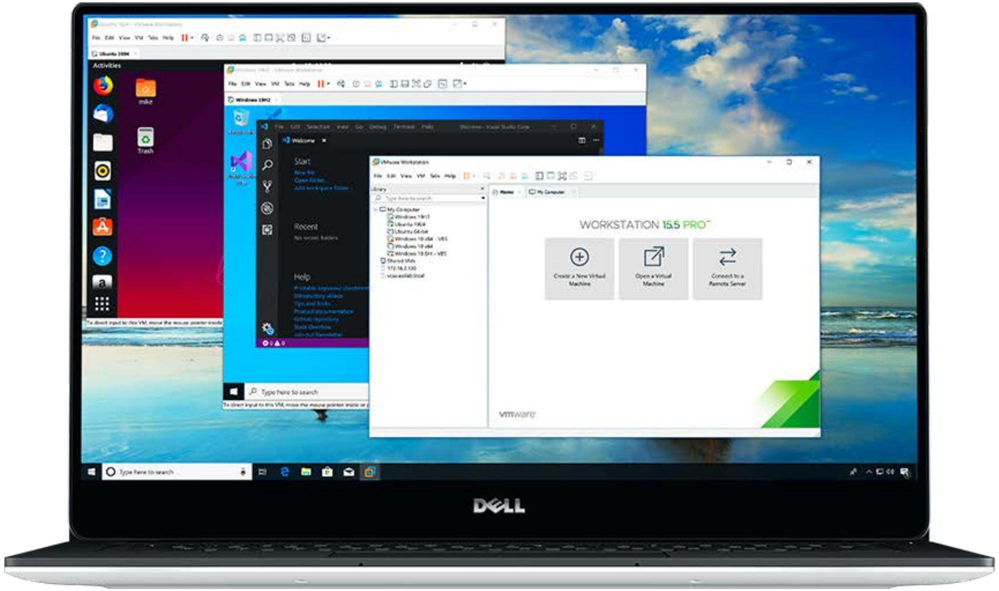


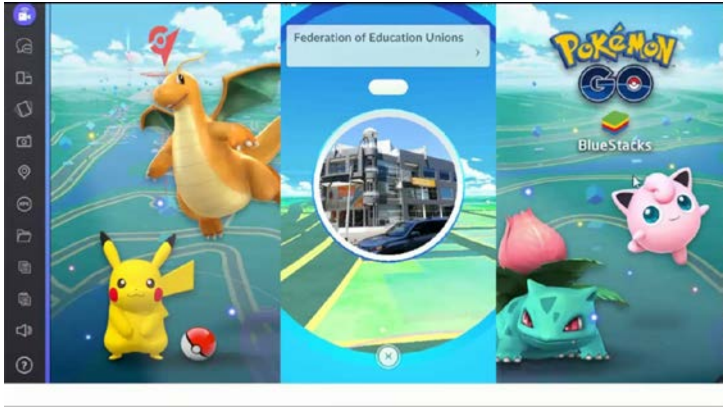
```
  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {$
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

If You Do Not Have Embedded Devices

- Virtual machine is a good helper
    - A **virtual machine**, known as a guest, is created within a computing environment, called a host.

Why Virtual Machine





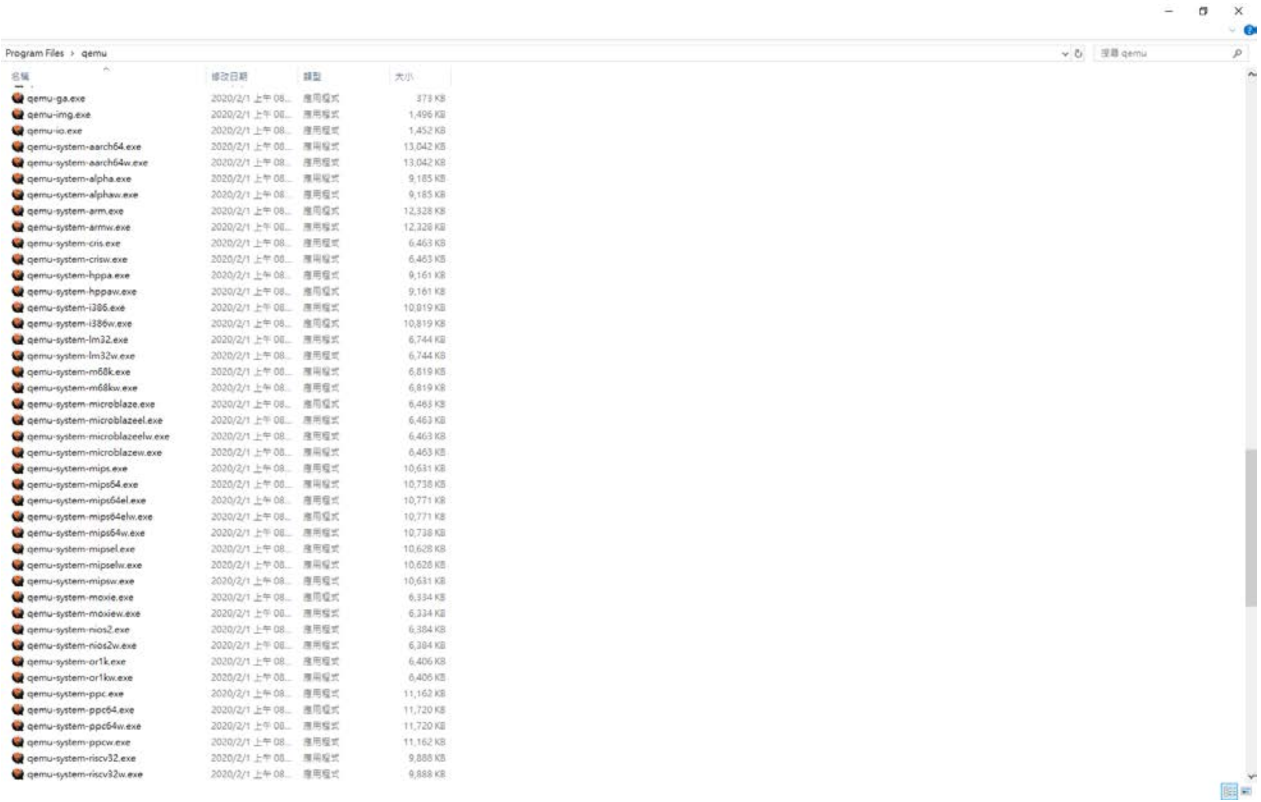
Virtual Machine

- System VM
  - Provide an environment for executing an OS.
  - Does hardware virtualization

- Process VM
  - Executing a single process

Why QEMU

Only for X86 Platform



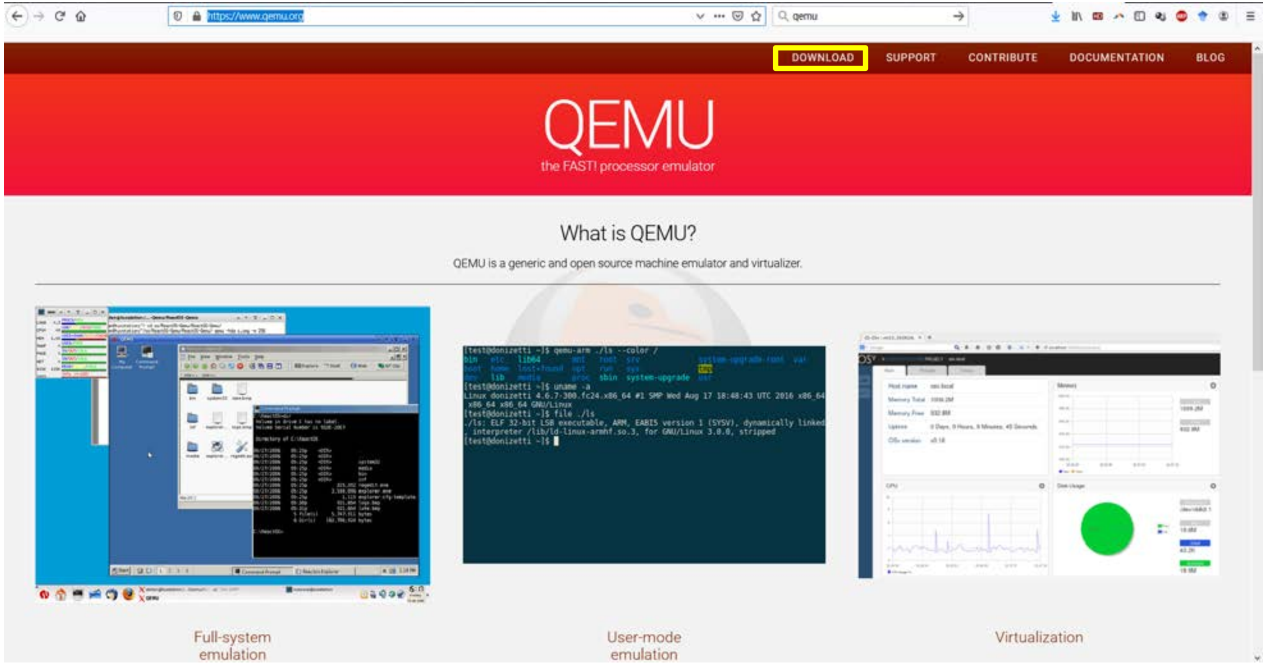
Program Files › qemu

搜尋 qemu

名稱	修改日期	類型	大小
qemu-ga.exe	2020/2/1 上午 08...	應用程式	373 KB
qemu-img.exe	2020/2/1 上午 08...	應用程式	1,496 KB
qemu-io.exe	2020/2/1 上午 08...	應用程式	1,452 KB
qemu-system-aarch64.exe	2020/2/1 上午 08...	應用程式	13,042 KB
qemu-system-aarch64w.exe	2020/2/1 上午 08...	應用程式	13,042 KB
qemu-system-alpha.exe	2020/2/1 上午 08...	應用程式	9,185 KB
qemu-system-alphaw.exe	2020/2/1 上午 08...	應用程式	9,185 KB
qemu-system-arm.exe	2020/2/1 上午 08...	應用程式	12,328 KB
qemu-system-armw.exe	2020/2/1 上午 08...	應用程式	12,328 KB
qemu-system-cris.exe	2020/2/1 上午 08...	應用程式	6,463 KB
qemu-system-crisw.exe	2020/2/1 上午 08...	應用程式	6,463 KB
qemu-system-hppa.exe	2020/2/1 上午 08...	應用程式	9,161 KB
qemu-system-hppaw.exe	2020/2/1 上午 08...	應用程式	9,161 KB
qemu-system-i386.exe	2020/2/1 上午 08...	應用程式	10,819 KB
qemu-system-i386w.exe	2020/2/1 上午 08...	應用程式	10,819 KB
qemu-system-lm32.exe	2020/2/1 上午 08...	應用程式	6,744 KB
qemu-system-lm32w.exe	2020/2/1 上午 08...	應用程式	6,744 KB
qemu-system-m68k.exe	2020/2/1 上午 08...	應用程式	6,819 KB
qemu-system-m68kw.exe	2020/2/1 上午 08...	應用程式	6,819 KB
qemu-system-microblaze.exe	2020/2/1 上午 08...	應用程式	6,463 KB
qemu-system-microblazeel.exe	2020/2/1 上午 08...	應用程式	6,463 KB
qemu-system-microblazeelw.exe	2020/2/1 上午 08...	應用程式	6,463 KB
qemu-system-microblazew.exe	2020/2/1 上午 08...	應用程式	6,463 KB
qemu-system-mips.exe	2020/2/1 上午 08...	應用程式	10,631 KB
qemu-system-mips64.exe	2020/2/1 上午 08...	應用程式	10,738 KB
qemu-system-mips64el.exe	2020/2/1 上午 08...	應用程式	10,771 KB
qemu-system-mips64elw.exe	2020/2/1 上午 08...	應用程式	10,771 KB
qemu-system-mips64w.exe	2020/2/1 上午 08...	應用程式	10,738 KB
qemu-system-mipsel.exe	2020/2/1 上午 08...	應用程式	10,628 KB
qemu-system-mipselw.exe	2020/2/1 上午 08...	應用程式	10,628 KB
qemu-system-mipsw.exe	2020/2/1 上午 08...	應用程式	10,631 KB
qemu-system-moxie.exe	2020/2/1 上午 08...	應用程式	6,334 KB
qemu-system-moxiew.exe	2020/2/1 上午 08...	應用程式	6,334 KB
qemu-system-nios2.exe	2020/2/1 上午 08...	應用程式	6,384 KB
qemu-system-nios2w.exe	2020/2/1 上午 08...	應用程式	6,384 KB
qemu-system-or1k.exe	2020/2/1 上午 08...	應用程式	6,406 KB
qemu-system-or1kw.exe	2020/2/1 上午 08...	應用程式	6,406 KB
qemu-system-ppc.exe	2020/2/1 上午 08...	應用程式	11,162 KB
qemu-system-ppc64.exe	2020/2/1 上午 08...	應用程式	11,720 KB
qemu-system-ppc64w.exe	2020/2/1 上午 08...	應用程式	11,720 KB
qemu-system-ppcw.exe	2020/2/1 上午 08...	應用程式	11,162 KB
qemu-system-riscv32.exe	2020/2/1 上午 08...	應用程式	9,888 KB
qemu-system-riscv32w.exe	2020/2/1 上午 08...	應用程式	9,888 KB

How to Get QEMU

- ➢ Go to https://www.qemu.org/



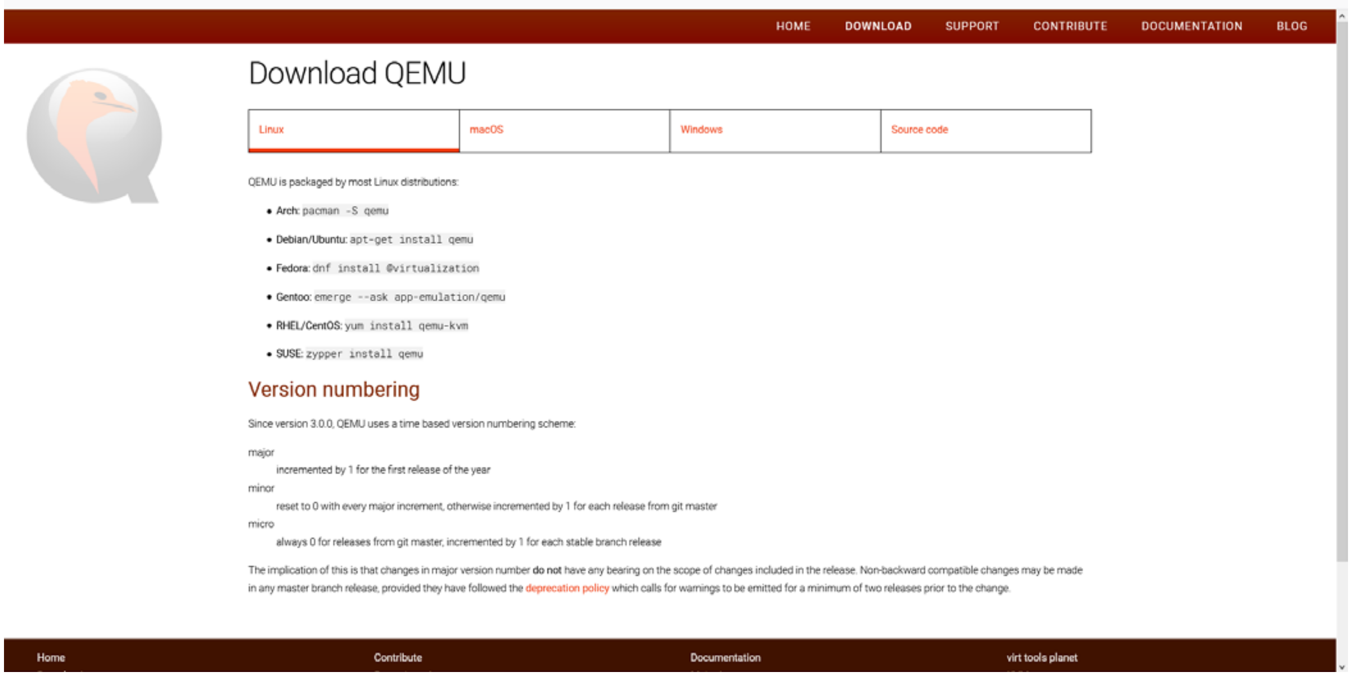
What is QEMU?

QEMU is a generic and open source machine emulator and virtualizer.

```
[test@donizetti ~]$ qemu-arm ./ls --color /
bin   etc   lib64        mnt   root  srv             system-upgrade-root  var
boot  home  lost+found   opt   run   sys             tmp
dev   lib   media        proc  sbin  system-upgrade  usr
[test@donizetti ~]$ uname -a
Linux donizetti 4.6.7-300.fc24.x86_64 #1 SMP Wed Aug 17 18:48:43 UTC 2016 x86_64
 x86_64 x86_64 GNU/Linux
[test@donizetti ~]$ file ./ls
./ls: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked
, interpreter /lib/ld-linux-armhf.so.3, for GNU/Linux 3.0.0, stripped
[test@donizetti ~]$
```

How to Get QEMU

- Go to https://www.qemu.org/



Download QEMU

QEMU is packaged by most Linux distributions:

- **Arch:** `pacman -S qemu`
- **Debian/Ubuntu:** `apt-get install qemu`
- **Fedora:** `dnf install @virtualization`
- **Gentoo:** `emerge --ask app-emulation/qemu`
- **RHEL/CentOS:** `yum install qemu-kvm`
- **SUSE:** `zypper install qemu`

Version numbering

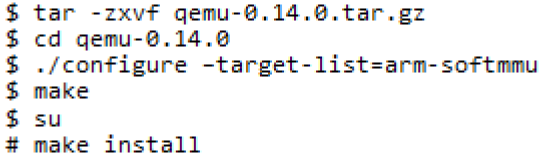
Since version 3.0.0, QEMU uses a time based version numbering scheme:

major
    incremented by 1 for the first release of the year
minor
    reset to 0 with every major increment, otherwise incremented by 1 for each release from git master
micro
    always 0 for releases from git master, incremented by 1 for each stable branch release

The implication of this is that changes in major version number **do not** have any bearing on the scope of changes included in the release. Non-backward compatible changes may be made in any master branch release, provided they have followed the deprecation policy which calls for warnings to be emitted for a minimum of two releases prior to the change.

Using QEMU for Embedded Systems

- ➢ Thank to the QEMU developers
  - ✓ Compiling QEMU for ARM



```
$ tar -zxvf qemu-0.14.0.tar.gz
$ cd qemu-0.14.0
$ ./configure -target-list=arm-softmmu
$ make
$ su
# make install
```

  - ✓ You will have two output binaries: `qemu-arm` and `qemu-system-arm`
    - `qemu-arm` is for ARM binary file execution
    - `qemu-system-arm` is to boot the ARM OS
  - ✓ Get ARM tool-chain from here
  - ✓ Refer to this website for more…

Summary

- In this lecture, we learn …
  - ✓ The definition of embedded O.S.
  - ✓ Some real-time scheduling methods
  - ✓ The system architecture of embedded O.S.
  - ✓ A brief introduction to Android system
  - ✓ How to get started with embedded system