



# *WinCE 与 ARM*

[www.farsight.com.cn](http://www.farsight.com.cn)

1. WinCE系统概述及应用
2. WinCE系统技术特点概述
3. WinCE系统与其它主流操作系统的区别
4. Bootloader的基本架构和流程

## Windows CE系统概述

1. 良好的可裁剪性和可移植性
2. 具备足以满足绝大多数应用场合的实时性
3. 与Win32 API的良好兼容性，包括多语言、DirectX等的支持
4. 丰富的应用软件支持，包括对通信，网络和多媒体等的支持

## 良好的可裁剪性和可移植性

- ✓ Windows CE的最小可执行内核大小约为200K，典型的内核大小为8M-20M左右
- ✓ 组件可以灵活的增减，开发环境会自动处理它们之间的依赖性
- ✓ Windows CE目前支持大量的主流嵌入式CPU如X86, MIPS, ARM, SuperH
- ✓ 提供了产品级BSP支持，最大限度的减少移植时间

## Windows CE的实时性

- ✓ 判断标准：最差响应时间，平均响应时间
- ✓ WinCE实时性的设计目标
- ✓ WinCE的实时性基于以下几个方面
  - ∅ 采用抢占式多任务内核
  - ∅ 支持嵌套中断，高优先级中断优先执行
  - ∅ ISR，IST机制
  - ∅ 强大的进程线程机制

## 与桌面Windows的良好兼容性

- √ 实现了Win32 API的子集
- √ 提供了MFC, ATL等模板支持
- √ 提供了.NET Framework的支持
- √ COM/COM+, Win Socket等大量与桌面Windows相兼容的技术
- √ 提供了多语言支持
- √ 通过ActiveSync等方式方便地与PC连接

## 丰富的应用软件支持

- √ 提供了IE, MSN, MS Office, Windows Media Player等大量的应用软件支持
- √ 提供了大量的应用支持库如VoIP支持, 各类多媒体编、解码器
- √ 强大的IDE和调试工具, 多种模拟器, 帮助缩短产品的上市时间

# Windows Embedded家族产品介绍

## √ Windows Embedded

### ∅ Windows Embedded CE

#### ü Windows Embedded CE

§ Core

§ Pro(增加Media、IE和更丰富的GUI组件)

§ Pro+(增加Office相关组件)

#### ü Windows Mobile

§ SmartPhone

§ Pocket PC

#### ü Windows AutoMotive

### ∅ Windows XP Embedded

## 各类WinCE产品与WinCE的区别

- ✓ 更丰富的组件，针对不同应用场合提供了更强大的支持
- ✓ 不同产品规定了相应的最小组件集，这些组件是必须包含在最终产品之内的

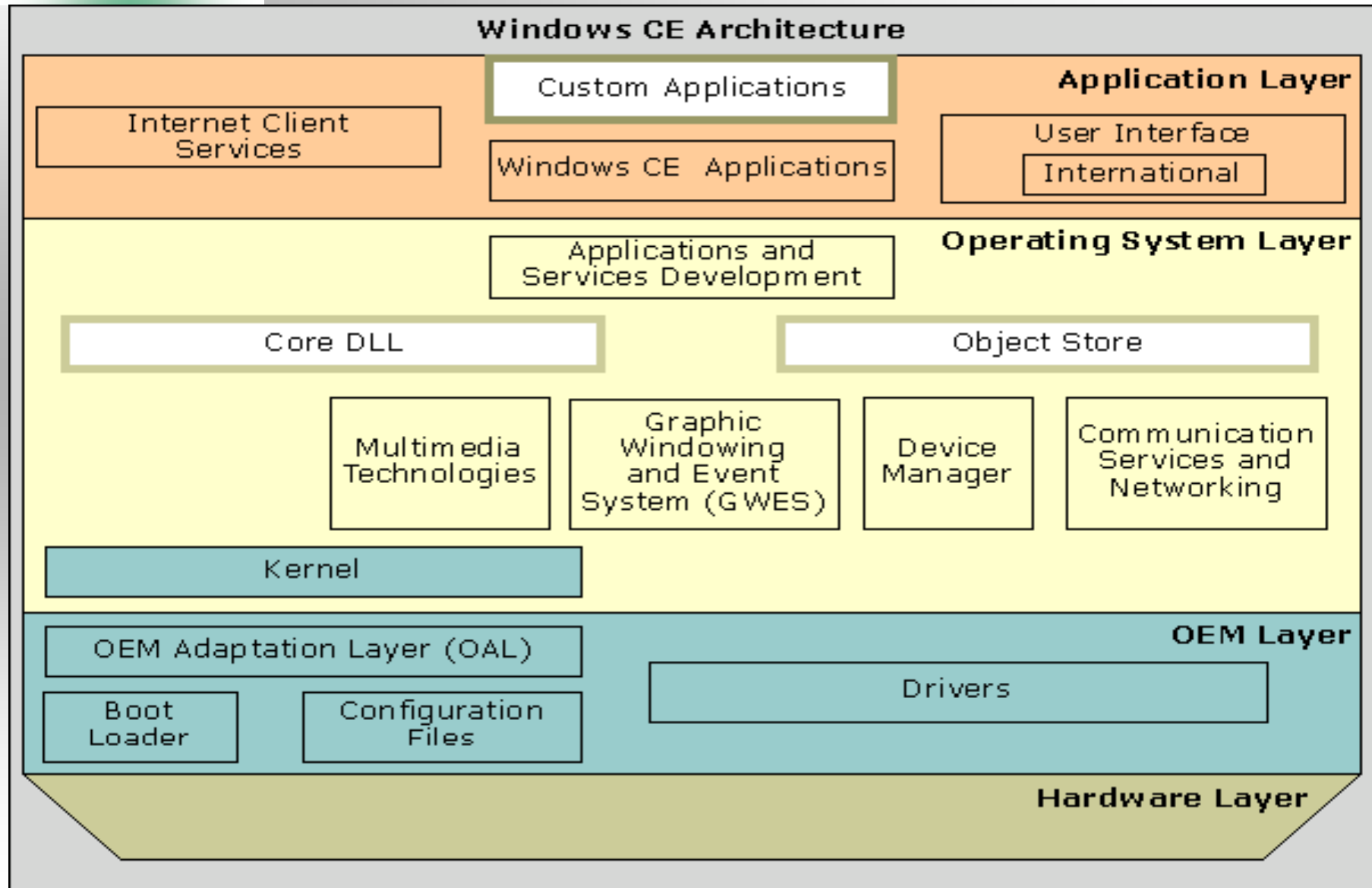
## 各产品的主要区别

- ✓ Windows Embedded CE 是基础版本
- ✓ Windows Mobile 包含Smartphone 和 Pocket PC 两个平台，专为要求特殊硬件配置的设备而设计。包含标准化的接口和应用程序，可确保在各种硬件设计中的兼容性。
- ✓ Windows Automotive 在共享 Windows CE 的丰富平台和服务的同时，还提供语音识别技术、自动化电源管理功能以及其他自动化特定的组件。

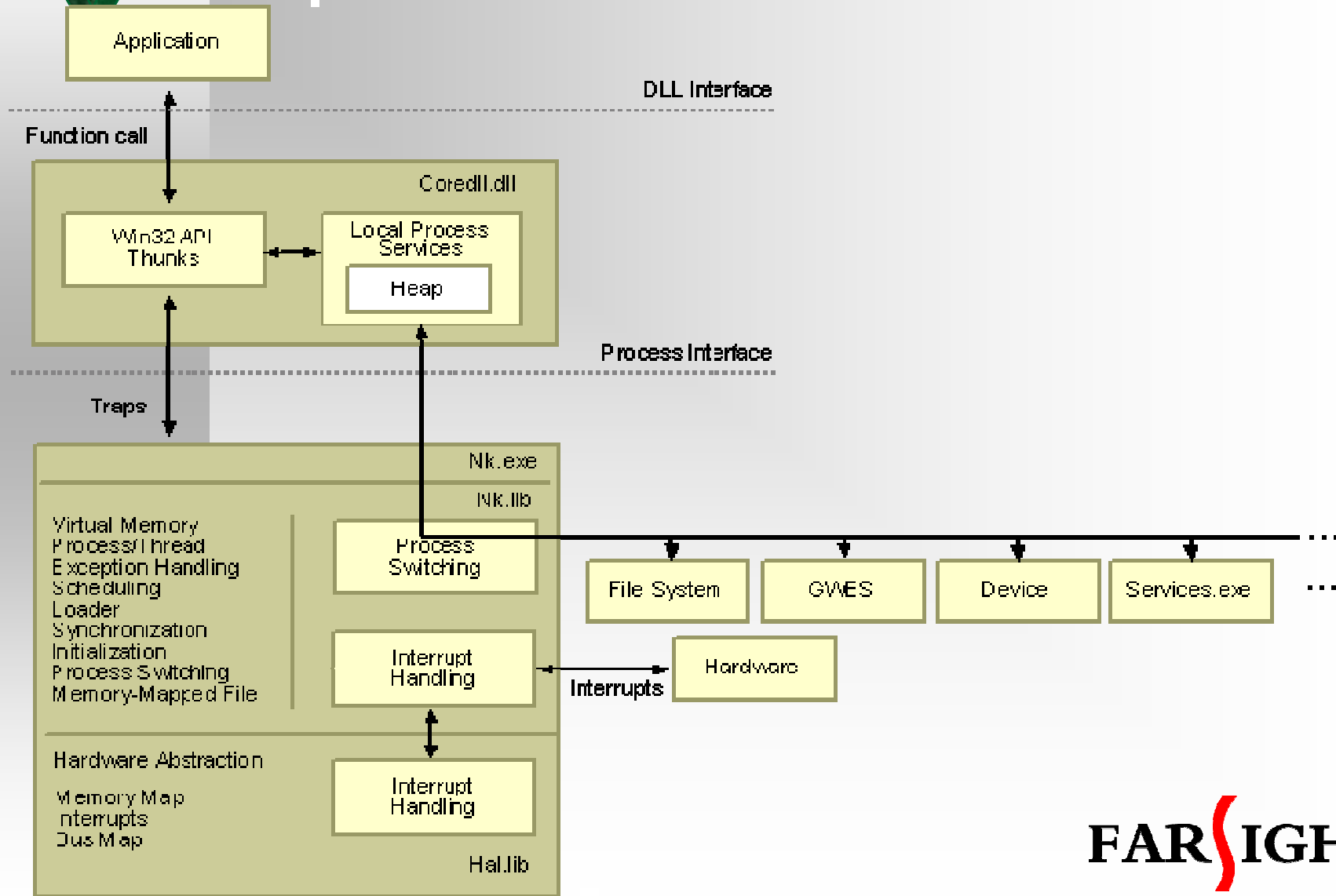
## Windows CE技术特点

1. 层次化的体系结构
2. 线程进程模型
3. 内存管理模型
4. 驱动管理与系统服务
5. 存储管理与文件系统
6. GUI
7. 应用程序

# 层次化的体系结构



# 层次化的体系结构



# 进程线程模型

- ✓ 进程
- ✓ 线程
- ✓ 调度
- ✓ 同步
- ✓ 进程间通信

- ✓ 进程是程序的运行实例，一个进程中至少包含一个线程
- ✓ WinCE系统中进程并不参与调度，也没有优先级和上下文
- ✓ 各进程的虚拟内存空间是独立的
- ✓ WinCE 5.0系统最多允许32个进程同时运行

- ✓ 线程是系统调度的最小单位，进程中的所有线程共享该进程中的资源。线程还拥有独立的堆栈等资源。
- ✓ 线程可以运行在核心态或用户态，运行在核心态时可以访问高端地址 ( $>0x80000000$ )
- ✓ 线程的数量只受系统资源的限制

- ✓ 作为抢占式多任务操作系统，WinCE采用基于优先级的时间片轮转算法调度线程
- ✓ 线程优先级可以从0到255
- ✓ 优先级反转与优先级继承

## ✓ 可以跨进程

- ∅ Mutex (互斥体)
- ∅ Semaphore (信号量)
- ∅ Event (事件)

## ✓ 不能跨进程

- ∅ Critical Section (临界区)
- ∅ Interlocked Functions (互锁函数)

## 进程间通信

- ✓ 方法有很多种，常用的是点对点消息队列和内存映射文件
- ✓ 内存映射文件的虚拟地址在0x42000000到0x7FFFFFFFFF的范围内
- ✓ 点对点消息队列是先入先出的buffer队列，可以传递任意大小的数据，有很大灵活性
- ✓ 点对点消息队列意味着它无法进行广播

## 驱动管理与系统服务

✓ WinCE中所有驱动都是用户态下的  
DLL

∅ 允许动态加载卸载驱动程序

∅ 增强稳定性

∅ 可以使用包括Win32 API在内的大量系统资源

✓ 所有的驱动程序DLL都是被  
Device.exe、GWES.exe或者  
FileSys.exe加载的

✓ GWES -DDI-> MDD -DDSI -> PDD

✓ GWES -DDI-> Monolithic Drivers

**FAR** **RIGHT**

## 其他技术特点

### √ 对象存储

∅ 注册表

∅ 文件系统

∅ 数据库

### √ GUI

### √ 应用程序

# Windows CE的内存管理模型

## √ 内存地址空间的划分与管理

- ∅ 内核空间与用户空间
- ∅ Slot的划分

## √ 内存的分配与使用

- ∅ malloc/free与new/delete
- ∅ 保留和提交
- ∅ Memory Map File

## √ 访问物理地址

# 内存地址空间的划分与管理

- ✓ 所有XIP DLL都会被映射到Slot1中
- ✓ Slot2一般是映射为FileSye.exe
- ✓ 包括GWES, Device和服务系统进程都会占Slot2-32中的某些Slot

Kernel space

0xFFFF FFFF	Kernel addresses: KPAGE, Trap area, others
0xE000 0000	Statically mapped virtual addresses: OEM additional
0xC400 0000	Slot 97: Nk.exe (Secure slot)
0xC200 0000	Unused
0xC000 0000	Statically mapped virtual addresses: UNCACHED
0x4000 0000	Statically mapped virtual addresses: CACHED
0x8000 0000	

User space

0x7FFF FFFF	Slot 63: Resource mappings
0x7E00 0000	Slots 33-62: Object store and memory mapped files
0x4200 0000	Slots 2-32: Processes
0x0400 0000	Slot 1: XIP dlls
0x0200 0000	Slot 0: Current process
0x0000 0000	

# 内存地址空间的划分与管理

- ✓ 每个进程的地址空间为64M
- ✓ XIP DLLs被加载到高地址的32M空间内

其中全局变量会被加载到低地址的32M空间内

- ✓ Code, Data和RAM DLLs被加载到低地址的32M空间内

应用程序无法在32M空间外动态分配内存

- ✓ 纯资源DLL不会被加载到当前进程的地址空间

## 内存的动态分配与使用

- √ 临时变量，是在栈中分配的
- √ 可以使用C/C++ Runtime Library中的内存分配函数malloc/free与new/delete，这些内存是在Heap中分配的
- √ 保留和提交(VirtualAlloc, VirtualFree等)分配虚拟内存（保留以64KB为单位，提交以页大小为单单位，如4KB）
- √ Memory Map File也可用来分配大内存，可以被所有进程访问

## 访问物理地址

- ✓ 先用 VirtualAlloc 保留一段虚拟地址，然后用 VirtualCopy 将虚拟地址与物理地址绑定
- ✓ 推荐使用 MmMapIoSpace 函数，它的原型就是使用上述两个函数实现的
- ✓ AllocPhysMem 用来分配一段物理上连续的内存，一般 DMA buffer 需要这样的内存

## WinCE 6.0在内存管理方面的增强

- ✓ 每个进程有2GB的虚拟地址空间，同时去掉了原来的共享内存区域
- ✓ 最多可以有32000个进程
- ✓ 用于加载DLL的空间增加到512M
- ✓ Device, FileSys, GWES等进程全部放入内核地址空间中执行
- ✓ 每个进程中的句柄不能被其他进程使用
- ✓ 采用了与桌面Windows相同的C运行时库

## WinCE与其它嵌入式系统比较

### √ WinCE

- ∅ 软实时
- ∅ 丰富的组件支持
- ∅ 第三方软件支持较多
- ∅ 适合于有复杂GUI的应用

## WinCE与其它嵌入式系统比较

### √ Linux

- ∅ 非实时，2.6以后对此部分有所增强
- ∅ 如果需要高实时性，有一些分支版本
- ∅ Open Source
- ∅ 有丰富的组件
- ∅ 第三方资源很多
- ∅ 适合于较复杂的支持多种协议的网络设备等

## WinCE与其它嵌入式系统比较

### √ VxWorks

- ∅ 硬实时，实时性强
- ∅ 组件较多，偏重于协议方面
- ∅ 第三方资源少
- ∅ 价格高

# Bootloader基本架构和流程

1. Boot Loader的基本流程
2. 硬件初始化的基本流程
3. EBoot的架构

## Boot Loader的基本流程

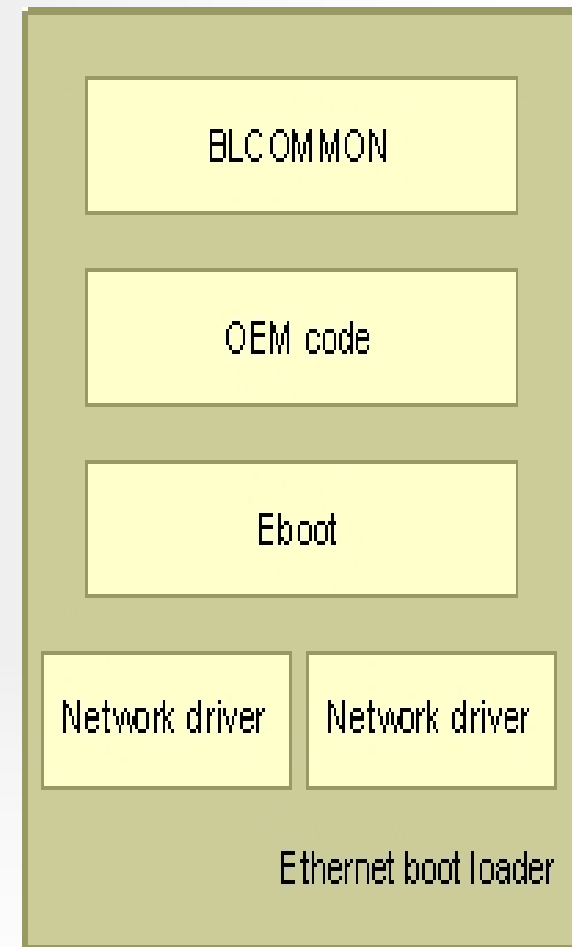
- √ 初始化硬件，包括CPU 状态，时钟，RAM
- √ 初始化堆栈，初始化外设，主要是调试和人机接口如串口，下载接口如网口和USB口等等
- √ 根据用户指令，执行不同的动作，如跳转到OS镜像、下载OS镜像、擦写Flash、修改默认参数等
- √ 可能会有一些特殊功能，如初始化LCD等

## 硬件初始化的基本流程

- ✓ 关闭WatchDog和中断
- ✓ 关闭MMU，清除Cache
- ✓ 配置时钟和PLL
- ✓ 配置DRAM控制器，并将RAM清零
- ✓ 将自身搬移到RAM中
- ✓ 设置栈指针SP
- ✓ 跳转到C语言代码
- ✓ 初始化各外设

- ✓ BLCommon提供通用的BootLoader架构，一般情况下不需要修改
- ✓ 用户实现如 OEMPlatformInit、OEMDebugInit等回调函数
- ✓ 必要的时候也可以自行修改 BLCommon

## EBoot的架构



华清远见

# Questions!



FAR  SIGHT

The logo features the word "FARSIGHT" in white, serif, all-caps font. A red, stylized vertical line separates the "FAR" and "SIGHT" parts. The text is centered within a dark green, textured, downward-pointing triangle that has a 3D effect with a lighter green top face and darker green side faces.

FARSIGHT

The success's road

[www.farsight.com.cn](http://www.farsight.com.cn)

谢谢！