基于 Yocto Project 的嵌入式应用设计

任宁宁，杨斌

（西南交通大学 信息科学与技术学院，成都 610031）

摘要：Yocto Project 是一个开源项目，可以在任意硬件上（如 x86、x86-64、ARM、PPC、MIPS）为任意嵌入式设备构建自定义嵌入式 Linux 应用所需的模板、工具和资源，极大地简化了开发过程。本设计主要基于 Yocto Project 在嵌入式设备上轻松定制嵌入式 Linux 应用，并实现 Yocto Project 的定制过程。

关键词：Linux；Yocto Project；定制系统

中图分类号：TP316　　**文献标识码**：A

Embedded Application Design Based on Yocto Project

Ren Ningning, Yang Bin

(School of Information Science & Technology, Southwest Jiao ong University, Chengdu 610031, China)

Abstract: Yocto Project is an open source project, and it can be applied in any hardware (such as x86, x86ō64, ARM, Power PC, MIPS) to build templates, tools and resources needed by customized embedded Linux application for any embedded device, and greatly simplify the development process. Embedded Linux application is easily customized on embedded device based on Yocto Project, and the customization process of Yocto Project is achieved.

Key words: Linux; Yocto Project; customized system

引　言

Linux 作为嵌入式系统的主要工具，具有源代码开放、完全可定制、支持许多网络协议、服务器级别高、可靠性高等很多优点。但现有的嵌入式 Linux 市场开始分化，现有的选择包括半导体厂商提供的实例方案、嵌入式 OSV 提供商应用的嵌入式 Linux 产品、嵌入式产品开发商的自有方案和开源项目等。由于缺乏一致性，造成了嵌入式开发高昂的维护成本，不仅嵌入式开发缺乏足够的专业人员，而且开发过程存在安全漏洞问题。在 Yocto Project 项目中，可以使用许多高效的工具，从而轻松定制嵌入式 Linux 产品。

1　Yocto Project 的原理和架构

Yocto Project 是一个开源项目，由 Linux Foundation 主导并被嵌入式业界领导者所支持。Yocto Project 与 OpenEmbedded 兼容，后者包括了许多开源项目的构建方法，可以作为 Yocto Project 的有效补充。Yocto Project 提供基于社区测试的支持多种架构的镜像。Yocto Project 的优点如下：具有高质量的构建系统，平等地支持所有主流的嵌入式架构（ARM、Power PC、MIPS、x86（32 & 64 位）），紧密跟踪许多上游开源项目的最新发布版本，具有统一的 Linux BSP 格式和应用程序开发套件，还可轻松地实现从原型切换到商用嵌入式 Linux 产品。

Yocto Project 架构中用户自定义层由用户根据需求定制，包括定制层、BSP（Board Support Package）层、特性层和核心元数据（Core Recipe）层。上层开源组建最常用的命令为 BusyBox 和 Eglibc 等。架构中应用的开发套件，由 Poky 构建系统生成交叉工具链，使用 ADT installer 来定制交叉开发环境，可以支持在目标系统和 sysroot 上的包管理架构，包括 Eclipse/Anjuta 集成开发环境的插件、调用安装好的交叉工具链，以及模板和调试工具。元数据准备好了由元工具（bitbake）解析、执行。元工具抽象构建一个软件过程中的 fetch、unpack、patch、configure、compile、package 等任务；同时，元工具还负责按不同软件之间的依赖关系有序地执行编译。编译成功后，用户自定义的 Yocto 镜像就可以在嵌入式设备上使用。Yocto Project 架构图如图 1 所示。

2　构建嵌入式系统

构建嵌入式系统是一个嵌入式 Linux 项目的核心，构建系统需要定义一组语义让开发者可以描述一个特定的

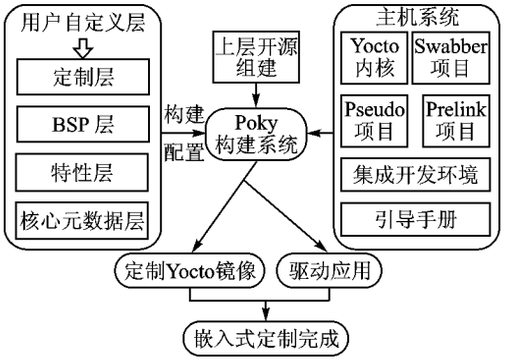

图 1　Yocto Project 架构图

构建任务，嵌入式系统负责交叉编译一个项目的整个过程，包括下载源代码、打上特定的补丁、配置、编译、安装，并将安装结果按指定格式打包。构建的嵌入式系统还要并发调度多个构建任务。好的构建系统决定了一个嵌入式 Linux 开发环境的效率和友好度。Yocto 内核使用 GIT 进行源代码管理，内核功能被组织成小的集合，方便深入定制化并支持多种内核开发流程和管理技术。所使用的应用开发套件是由 Poky 构建系统生成的交叉工具链，使用 ADT installer 来定制交叉开发环境，支持在目标系统和 sysroot 上的包管理架构，包括 RPM、DEB、IPK，支持 Eclipse/Anjuta 集成开发环境的插件，而且还可以使用 Qemu 作为模拟器，模拟器支持基于 OpenGL 的应用开发（QemuGL 加速）。

(1) 元数据 Recipe

一份 Recipe 包括了一组元数据，而这些元数据则定义了与一个项目相关的构建信息：源代码的地址，项目相关的特定配置参数，如何编译、安装和打包数据。丰富的类文件有助于最大程度地重用元数据，由 Profile 决定选择哪些 Recipe。Yocto Project 包括了一些事例 Profile，一份完整工作的 Recipe 可能只需要 3 行元数据。

(2)元数据层

元数据可以相互层叠，以很低的维护成本来进行深层次的定制化，同时还可以增加新的项目，调整架构相关的编译标志，覆盖某个项目特定的配置选项。

开发者提供元数据（食谱或菜谱），食谱是由一种标记性的语言所写，由元工具（bitbake）解析、执行。每份食谱都是描述某个软件的“清单”，例如：源代码从哪里下载，需要哪些补丁文件，可能需要哪些特殊编译链接选项，打包时需要的特殊配置，依赖哪些其他软件等相关信息。元工具则抽象了构建一个软件过程中的 fetch、unpack、patch、configure、compile、package 等任务；同时，元工具还负责按不同软件之间的依赖关系有序地执行编译。

3　智能车载系统硬件设计

随着汽车电子的发展，智能车载系统越来越受到关注，智能车载系统通过信息的传递为车辆提供出行指引、安防、救援、远程故障诊断等服务。远程信息服务（Telematics）即通信网络为安装在车上的资讯系统平台提供的多样化的信息服务。Telematics 系统可分为车前座系统、车后座系统，以及车况诊断系统。

其中，车前座系统提供的服务包括通信、导航、行车安全监视、联网资讯、路况、天气等；车后座系统的服务包括在线下载影音资讯、在线网络游戏等；车况诊断系统的服务包括保养通知、车况预警等。目前，在车载智能系统中有日产的 Carwings、通用公司的 Onstar、丰田公司的 G-BOOK、一汽奔腾公司的 D-Partner 以及荣威公司的 inKanet 等。其中，OnStar 是以安防为主的 Telematics 服务，G-Book 是以资讯及娱乐为主的 Telematics 服务。

Telematics 服务功能如下：

① 卫星定位：通过 GPS 配合路线资讯，作路况报道与路线指引；

② 道路救援：行车过程中，假使发生车祸或故障意外，通过按键自动联系救援；

③ 汽车防窃：通过卫星定位提供失窃车辆的搜寻与追踪，并短信通知车主；

④ 自动防撞系统：通过传感器或雷达，感应车与车间的安全行驶距离；

⑤ 车况掌握：车辆性能与车况的自动侦测、维修诊断等；

⑥ 个人化资讯接收：收发电子邮件与个人化资讯等；

⑦ 多媒体娱乐资讯：高画质与高音质的视听设备、游戏机、上网机、个人资讯中心随选视讯等。

⑧ 拖车追踪：是一种追踪通过安装在拖车的翼卡车联网和移动通信网络或卫星通信定位数据的技术。

⑨ 紧急救援：车主在行车过程中如遇到车辆缺油、缺水、故障等现象，可通过按下紧急按钮向服务中心进行求救。

根据智能车载系统的功能需求，把车载系统的硬件结构以模块的形式实现，智能车载系统硬件结构框图如图 2 所示。

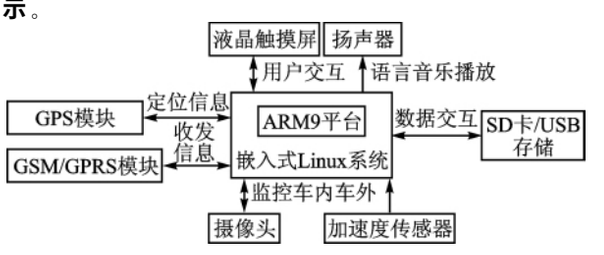

图 2　智能车载系统硬件结构框图

要实现卫星定位和道路紧急救援，需要通过 GPS 模块作路况报导与路线指引，道路救援和汽车防窃需要 GSM/GPRS 模块来通知车主和收发信息。多媒体娱乐资讯的用户交互需要液晶触摸屏和扬声器来实现，数据交互功能采用 SD 卡/USB 存储。摄像头监控车内外实时的情况，智能车载系统还需要加速度传感器来测量加速度。

4 智能车载系统软件设计和实现过程

采用 Yocto project 来定制智能车载系统的软件功能模块，如图 3 所示。智能车载系统的软件主要实现 GPS 模块、显示模块、控制模块、通信模块和娱乐模块等的功能等。

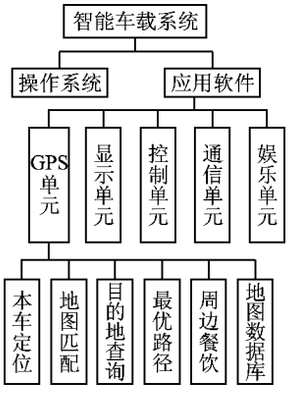

图 3 智能车载系统软件功能模块

4.1 定制智能车载系统平台

在 Yocto Project 中，现有的元数据以功能集合来划分，通过 Profile 来定义用户所需要的集合，智能车载系统需要的功能有 Wi-Fi、GPS、GPRS、USB、serial、keyboard、touchscreen 等，所以只需要在 Profile 文件中修改目录：

DISRO_FEATURES="alsa ext2 touchscreen wifi usb serial keyboard …"

4.2 智能车载系统中添加一个新的项目

添加一个新项目只需要 3 行元数据，需要指定依赖关系、指定源代码地址、继承类，然后会根据继承的 Yocto Project 类来自动生成所需要的构建任务。

首先，要将源代码从网上下载。bitbake 通过 SRC_URI 变量知道到哪里去下载源码。把需要的源代码地址指定：

SRC_URI=http://sourceforge.net/project/

4.3 定制实现过程

在 Linux 下安装好需要的包之后创建镜像：

```
$ wget http://www.yoctoproject.org/downloads/poky/poky-bernard-5.0.1.tar.bz2
$ tar xjf poky-bernard-5.0.1.tar.bz2
$ source poky-bernard-5.0.1/poky-init-build-env poky-5.0.1-build
```

用 bitbak 定制系统：

```
$ bitbake -k poky-image-sato
```

使用模拟器，模拟器中用户可使用终端：

```
$ poky-qemu qemux86
```

定制过程完成，用户可以定位车载、收发信息，并实时监控车内外情况以及享受听歌、播放视频、上网等智能车载系统功能。

结 语

采用 Yocto Project 构建的智能车载系统，实现了智能车载的信息定位、信息收发、数据交互、实时监控、用户交互，以及多媒体娱乐功能，构建过程灵活简单。构建一个嵌入式 Linux 系统需要构建引导模块、内核和文件系统。这是一个相当复杂的过程，特别是文件系统的构建。Yocto Project 就是为了简化嵌入式系统的构建过程而设计。由以上设计可以看出，Yocto Project 提供足够的灵活性，定制化非常简单，当用户不需要应用程序的可移植性，并且用户的嵌入式设备需要深度定制化时，使用 Yocto Project 最为省时省力。

参考文献

[1] Wind River Corporation. The Yocto Project Kernel Architecture and Use Manual[EB/OL]. [2012-04]. http://www.yoctoproject.org/documentation.

[2] Richard Purdie, Linux Foundation. Poky Reference Manual [EB/OL]. [2012-04]. http://www.yoctoproject.org/documentation.

[3] Intel Corporation. Board Support Package Developer's Guide [EB/OL]. [2012-04]. http://www.yoctoproject.org/documentation.

[4] Yocto Project Quick Start[EB/OL]. [2012-04]. http://www.yoctoproject.org/documentation.

[5] Jessica Zhang, Intel Corporation. Application Development Toolkit User's Guide[EB/OL]. [2012-04]. http://www.yoctoproject.org/documentation.

[6] 余刚，冯桑. 车载智能显示终端系统设计方案[J]. 广东大学学报，2010(6)：77-79.

任宁宁(研究生)，主要研究方向为嵌入式系统软件开发；杨斌(教授)，主要研究方向为单片机及嵌入式系统应用。

嵌入式资源免费下载

总线协议：

1. 基于 PCIe 驱动程序的数据传输卡 DMA 传输
2. 基于 PCIe 总线协议的设备驱动开发
3. CANopen 协议介绍
4. 基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计
5. FPGA 实现 PCIe 总线 DMA 设计
6. PCI Express 协议实现与验证
7. VPX 总线技术及其实现
8. 基于 Xilinx FPGA 的 PCIE 接口实现
9. 基于 PCI 总线的 GPS 授时卡设计
10. 基于 CPCI 标准的 6U 信号处理平台的设计
11. USB30 电路保护
12. USB30 协议分析与框架设计
13. USB 30 中的 CRC 校验原理及实现
14. 基于 CPLD 的 UART 设计
15. IPMI 在 VPX 系统中的应用与设计

VxWorks：

1. 基于 VxWorks 的多任务程序设计
2. 基于 VxWorks 的数据采集存储装置设计
3. Flash 文件系统分析及其在 VxWorks 中的实现
4. VxWorks 多任务编程中的异常研究
5. VxWorks 应用技巧两例
6. 一种基于 VxWorks 的飞行仿真实时管理系统
7. 在 VxWorks 系统中使用 TrueType 字库
8. 基于 FreeType 的 VxWorks 中文显示方案
9. 基于 Tilcon 的 VxWorks 简单动画开发
10. 基于 Tilcon 的某武器显控系统界面设计
11. 基于 Tilcon 的综合导航信息处理装置界面设计
12. VxWorks 的内存配置和管理
13. 基于 VxWorks 系统的 PCI 配置与应用

14. 基于 MPC8270 的 VxWorks BSP 的移植

Linux：

1. Linux 程序设计第三版及源代码
2. NAND FLASH 文件系统的设计与实现
3. 多通道串行通信设备的 Linux 驱动程序实现
4. Zsh 开发指南-数组
5. 常用 GDB 命令中文速览
6. 嵌入式 C 进阶之道
7. Linux 串口编程实例

Windows CE：

1. Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计
2. Windows CE 的 CAN 总线驱动程序设计
3. 基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究
4. 基于 Windows CE.NET 平台的串行通信实现
5. 基于 Windows CE.NET 下的 GPRS 模块的研究与开发
6. win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码
7. Windows 下的 USB 设备驱动程序开发
8. WinCE 的大容量程控数据传输解决方案设计
9. WinCE6.0 安装开发详解
10. DOS 下仿 Windows 的自带计算器程序 C 源码
11. G726 局域网语音通话程序和源代码
12. WinCE 主板加载第三方驱动程序的方法
13. WinCE 下的注册表编辑程序和源代码
14. WinCE 串口通信源代码
15. WINCE 的 SD 卡程序[可实现读写的源码]

PowerPC：

1. Freescale MPC8536 开发板原理图

2. 基于 MPC8548E 的固件设计
3. 基于 MPC8548E 的嵌入式数据处理系统设计
4. 基于 PowerPC 嵌入式网络通信平台的实现
5. PowerPC 在车辆显控系统中的应用
6. 基于 PowerPC 的单板计算机的设计
7. 用 PowerPC860 实现 FPGA 配置

ARM：

1. 基于 DiskOnChip 2000 的驱动程序设计及应用
2. 基于 ARM 体系的 PC-104 总线设计
3. 基于 ARM 的嵌入式系统中断处理机制研究
4. 设计 ARM 的中断处理
5. 基于 ARM 的数据采集系统并行总线的驱动设计
6. S3C2410 下的 TFT LCD 驱动源码
7. STM32 SD 卡移植 FATFS 文件系统源码
8. STM32 ADC 多通道源码
9. ARM Linux 在 EP7312 上的移植

Hardware：

1. DSP 电源的典型设计
2. 高频脉冲电源设计
3. 电源的综合保护设计
4. 任意波形电源的设计
5. 高速 PCB 信号完整性分析及应用
6. DM642 高速图像采集系统的电磁干扰设计
7. 使用 COMExpress Nano 工控板实现 IP 调度设备