

Linux 用 USB デバイスドライバの開発支援に関する一提案 A Development Support System for Linux USB Device Driver

城戸 英之† 吉田 泰彦† 大原 茂之‡
Hideyuki Kido Yasuhiko Yoshida Shigeyuki Ohara

1. はじめに

今日, USB は PC と周辺機器をつなぐインターフェースの標準になっており, ほとんどの PC, OS は USB をサポートしている. Linux はカーネル 2.4 より USB が正式にサポートされており, 今後 2.4 以降のカーネルが普及することで, 多くの PC は Linux 上で USB が使えるようになる.

新しい USB デバイスを利用する場合, ユーザはその機器のデバイスドライバを用意しなければならない. 多くの場合, デバイスドライバはデバイスのメーカーによって用意されている. しかし, メーカーは Windows や MAC 用の提供しか行っていないのが現状である. このことから Linux で USB デバイスを利用する場合, ユーザがその開発を行う必要がある. Linux 用 USB デバイスドライバの開発は, USB の知識が必要なだけでなく, LinuxUSB システムの知識が必要とされる. このため Linux 以外の OS 用の USB デバイスドライバの開発を行っていた開発者でも Linux 用 USB デバイスドライバの開発のためには新たに知識を習得する必要がある. それによって開発者に負担が掛かるだけでなく, 開発に時間を要する. そこで, USB デバイスドライバの開発の際に Linux のデバイスドライバや USB システムの知識を最小限のもの開発を行えるように支援することで, 開発は容易になると考えられる.

そこで本研究では, Linux 用 USB デバイスドライバの自動生成ツールの開発を行う. 開発者はこのツールを利用することで, 開発の負担を軽減し開発期間を短縮することができる.

2. LinuxUSB システムと自動生成対象

LinuxUSB システムの構成を Figure.1 に示す. USB デバイスドライバは USB コアダライバの USB デバイスドライバ用 API を呼び出しユーザプログラムからの要求を処理する. 自動生成ツールの生成対象は, 図中に示す USB デバイスドライバである.

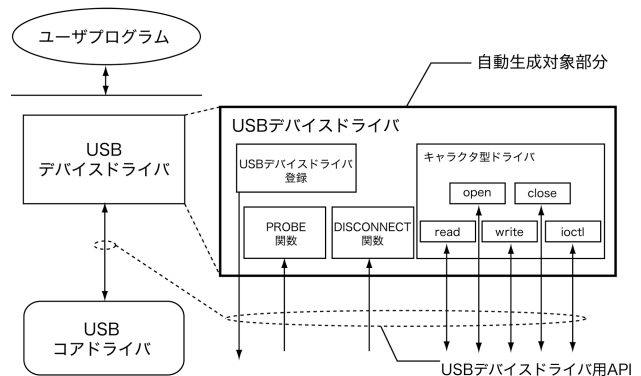


Figure.1 LinuxUSB システムと自動生成対象

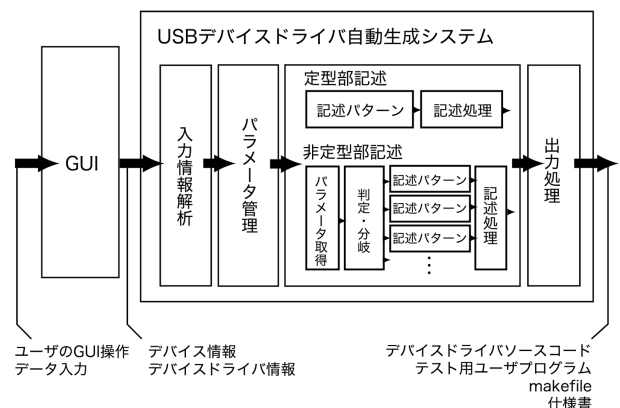


Figure.2 自動生成システム

3. Linux 用 USB デバイスドライバ自動生成システム

Linux 用 USB デバイスドライバ自動生成システムは, デバイスドライバ開発の対象となるデバイス情報・データの送受信の処理情報・Linux デバイスドライバの情報を入力として, デバイスドライバソースコード・makefile・テスト用ユーザプログラム・仕様書を出力する. ユーザは GUI を利用して入力を行い, その入力物から USB デバイスドライバ自動生成システムが出力物を生成する. 自動生成システムの構成を Figure.2 に示す.

各入力物の説明をする. デバイスドライバの開発対象となるデバイスの情報とは, Vendor ID, Product ID, 各エンドポイントの情報である. 各エンドポイントの情報とは, コンフィグレーション番号, インターフェース番号, オルターネート設定, エンドポイントアドレス, データの流れ (IN または OUT), データ転送形態, パケット

† 東海大学大学院工学研究科電子工学専攻

‡ 東海大学電子情報学部情報メディア学科教授

サイズである．データ送受信の処理情報とは，各エンドポイントとのデータをやり取りする際のデータの扱い方である．Linux デバイス処理情報とは，Linux のデバイスドライバに必要となる情報で，デバイスドライバの名前，バージョン，ライセンス，ディスクリプション，マイナ番号である．

各出力物の説明をする．USB デバイスドライバのソースコードは，デバイスドライバ開発対象のデバイスのデバイスドライバである．ソースコードは，デバイスドライバのコードとヘッダファイルである．テスト用プログラムとは，生成したデバイスドライバのテストを行うためのプログラムである．このプログラムは，デバイスドライバのデータ送受信のテストのみを行うプログラムである．Makefile は，USB デバイスドライバのソースコードとテスト用プログラムのソースコードをコンパイルするものである．仕様書は，自動生成の際の，システムの入出力の確認，USB デバイスドライバソースコードとテスト用プログラムのコンパイル方法を記述した HTML ファイルである．

3.1 デバイス情報の入力・自動取得

まず，ユーザはデバイス情報を入力する．情報の入力の方法はデバイスから情報を自動取得する方法と，デバイスの情報を手動入力する方法がある．対象デバイスがある場合には proc ファイルシステム（/proc/bus/usb/devices）から情報を取り出すことで自動的に取得することができる．対象デバイスがない場合は手動入力を行う．

次に，Linux デバイスドライバ情報を入力する．Linux デバイスドライバ情報には，デバイスドライバの名前やマイナ番号などユーザ環境を考慮する必要がある．そのため，ユーザが手動入力する．

3.2 データ送受信の処理情報の入力

次に，データ送受信の処理情報を入力する．データ送受信の処理情報はあらかじめ用意されているパターンより選択する．用意したパターンは各エンドポイントに適したデータの処理方法である．

Table.1 入力に対応する記述部分

入力	記述部分（一部）
・デバイス情報 Vendor ID, Product ID 各エンドポイント情報	初期化処理用構造体 通信関数内処理
・Linux デバイスドライバ情報 デバイスドライバの名前 マイナ番号	関数・変数の名前 初期化処理用構造体
・データ送受信の処理情報 データ送受信方法	通信関数内処理

3.3 デバイスドライバの記述

GUI 部で得られたデバイス情報とデータ送受信の処理情報をもとに，USB デバイスドライバ自動生成システムで出力物を生成する．まず，入力された情報は入力情報解析部において，生成に必要なパラメータが抜き出され，パラメータ管理部に格納される．以降の記述処理は，この格納された情報をもとに行われる．入力に対応する記述部分を Table.1 に示す．

デバイスドライバの記述方法を説明する．デバイスドライバは，構造体と関数内にエンドポイントのアドレス，通信方法やバッファサイズなどに応じた変数や処理から構成される．USB デバイスドライバ登録関数，open 関数，close 関数，disconnect 関数は関数名以外にデバイスに依存する部分が無いため，あらかじめ用意されている記述パターンをそのまま記述する．probe 関数ではデバイスを識別する処理を記述するため，デバイス情報をより記述を行う．ioctl 関数，送受信処理関数ではデバイスとユーザプログラム間のデータの送受信処理を記述するため，各エンドポイント情報，データ送受信処理方法の情報から記述を行う．記述を行うとき，パラメータ管理部より情報を取得し，それにあった記述パターンを選択し記述を行う．Figure.3 に記述例を示す．

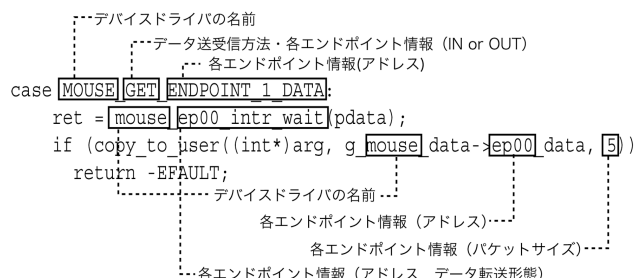


Figure.3 記述例

4. おわりに

本報告では，Linux 用 USB デバイスドライバ自動生成システムを提案した．今後の予定として，データ処理方法などを増やし，多くのデバイスに対応できるようにする．

参考文献

- [1] 山岡賢一，“Linux 用 USB ドライバの作成時例”，USB ハード & ソフト開発のすべて 272~278, CQ 出版社(2001)
- [2] ALESSANDRO RUBINI, JONATHAN CORBET 著，山崎康，山崎邦子，長原宏治，長原陽子 訳，“Linux デバイスドライバ 第 2 版”，オライリー・ジャパン (2002)