実技課題

実技課題

「Linuxのオペレーションと組込みLinuxのプログラミング」

|  |
| --- |
| １　作業時間（※）  　　 １２０分（開発環境等の仕様に応じて延長も可）  ２　配付資料  問題用紙  ３　課題作成、提出方法  課題は実機にて作成し、実機にて評価するため提出は不要 |

１．課題仕様

　課題の手順は以下のとおりとする。

1. 実習機のパソコン（以下ホストPC）に仮想環境（以下ゲストPC）にLinuxをクロス開発環境としてインストールを行なう。仮想環境の動作仕様は下記の表1に示す。
2. ゲストPCのLinuxに対し下記の表2の設定を行なう。設定はインストール時、またはインストール後に設定しても良いものとする。
3. ゲストPCに構築したLinuxに対し下記の表3の開発環境構築、表4プログラミングの作業を行なう。作成した課題は指定箇所に保存しておくこと。

表 ：環境構築の仕様例（ホストOS：Windows、ホストOS：Debianを想定）

|  |  |  |
| --- | --- | --- |
| **項目** | **設定等** | **備考** |
| Linuxのディストリビューション | Debian 7.6.0 | ターゲットに合わせたディストリビューションを選択 |
| ゲストPC名 | kadai |  |
| ゲストPCデータ  (ホストPCにおける保存先) | D:\Linux\kadai | ホストPCはWindowsを想定 |
| ゲストPCのディスクサイズ | 8GB以上 |  |
| ゲストPCのメモリサイズ | 512MB以上 |  |
| SDカードリーダ |  | ホストPC、ゲストPCで使用可能なもの |
| SDカード |  | 2GB以上を想定 |

表 ：環境構築の仕様例（ホストOS：Windows、ホストOS：Debianを想定）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項目** | | | **設定等** | **備考** |
| 言語、地域、キーボードの設定 | | | 日本語 | 日本語環境に適した設定を適用 |
| ネットワークの設定 | IPアドレス | | 192.168.0.xx/24 | xxには講師より指示された数値を指定する。必要に応じてプロキシサーバ等の設定も行なうこと。 |
| デフォルトゲートウェイ | | 192.168.0.1 |
| DNSサーバ | | 192.168.0.1 |
| ホスト名 | | | kadaixx |
| スーパー  ユーザ | | ユーザ名 | root | rootユーザ |
| パスワード | rootpass |  |
| 通常  ユーザ | | ユーザ名 | user |  |
| パスワード | user |  |

表 ：作業内容

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **作業番号** | **作業内容** | | | **備考** |
| １ | コマンドサーチパスPATHに"/sbin"を（継続的に）追加する。 | | |  |
| ２ | "testuser"という一般ユーザを追加する。 | | |  |
| ３ | "testuser"に"password"というパスワードを設定する。 | | |  |
| ４ | build-essential、linux-Headerをインストールする。 | | |  |
| ５ | IPアドレスを192.168.0.yyに変更する。 | | | yyは講師より指示された番号 |
| ６ | SDカードを接続し、マウントする。 | | |  |
| ７ | SDカードを以下のようにファイルシステム構築する。 | | |  |
| **領域** | **形式** | **容量** |  |
| 領域1 | FAT32 | 1GB |
| 領域2 | ext3 | 700MB |
| 領域3 | swap | 領域1, 2以外 |
| ９ | SDカードをアンマウントし、Linux領域のみマウントする。 | | |  |

２．課題

　１．表2のクロス開発環境を構築する。

　２．表3の作業を行なう。

　３．以下のハードウェア構成に対してプログラミングを行なう。

押しボタンスイッチSW0を押す(ONにする)たびにLED0が、点灯・消灯するプログラムを作成する。プログラムは以下をベースとすること。

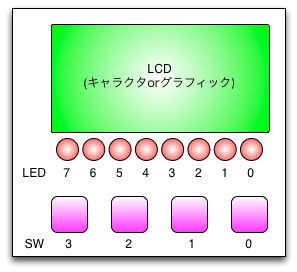


図 ：ターゲットボードのスイッチ(SW)とLED

|  |  |
| --- | --- |
| 行 | コード |
|  | #include <stdio.h> |
|  | #include <unistd.h> |
|  | #include <sys/wait.h> |
|  | #include <fcntl.h> |
|  | #include <linux/input.h> |
|  | #define SW "/dev/input/event0" |
|  | #define LED "/sys/class/leds/" |
|  |  |
|  | void killchild(int signum); |
|  | int i1,i2; |
|  |  |
|  | int main(int argc,char \*argv[]) |
|  | { |
|  | int fd; // デバイスドライバ用ファイルディスクリプタ |
|  | int pipe\_fd[2]; // パイプ用ファイルディスクリプタ |
|  | struct input\_event ev; |
|  | int dummy = 1; |
|  | unsigned char r=0; |
|  | char data[3]; |
|  | int st; |
|  | struct sigaction sig={0}; |
|  |  |
|  | sig.sa\_handler = killchild; // ハンドラ関数を設定 |
|  | sigaction(SIGINT, &sig, NULL); // シグナルINTに対するハンドラ関数の登録 |
|  |  |
|  | pipe(pipe\_fd); // パイプの生成 |
|  |  |
|  | if ((i1 = fork()) == 0) { // スイッチプロセス |
|  | fd = open(TP,O\_RDONLY); |
|  | while (1) { |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | } |
|  | } |
|  | close(fd); |
|  | } |
|  |  |
|  | if ((i2 = fork()) == 0) { // LEDプロセス |
|  | while (1){ |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | } |
|  | } |
|  | wait(&st); |
|  | killchild(0); |
|  | return 0; |
|  | } |
|  |  |
|  | void killchild(int signum) |
|  | { // プロセス終了 |
|  | kill(i1, SIGTERM); |
|  | kill(i2, SIGTERM); |
|  | \_exit(1); |
|  | } |