実習１：複数のプロセスから同時にLEDを制御 (1/3)

igel

- デバイスドライバの機能仕様
  - デバイスファイル「/dev/ledctrl0」経由のread()/write()でLEDの点灯／消灯を制御
  - read()は、unsigned short(u_short)型の値により全LEDの状態（＝レジスタ設定値）を出力
  - write()は、u_short型の値を入力し、全LEDの状態を更新
  - モジュールのロード／アンロード時、全LEDを消灯
- ユーザプログラムの機能仕様
  - 指定された1LEDを一定間隔で点滅
  - /dev/ledctrl0を介してLEDを制御
  - 点滅するLED（番号, 0-7）と点滅間隔（ミリ秒）をコマンドライン引数で指定
  - 点滅を256回繰り返した後終了

実習１：複数のプロセスから同時にLEDを制御 (2/3)

igel

- デバイスドライバの実装仕様
  - デバイス名は「ledctrl」、メジャー番号は240、マイナー番号は0
  - u_short型の変数にLED状態（レジスタ設定値）を保存
  - read()は上記変数値をcopy_to_user()で転送
  - write()は、sizeof(u_short)バイトの入力値をcopy_from_user()で受けて、レジスタに設定、同時に上記変数に設定値を保存
- ユーザプログラムの実装仕様
  - コマンドライン引数は、第1引数にLED番号（0-7）、第2引数に点滅間隔（ミリ秒）
  - 点滅間隔の調整は、nanosleep()を使用
  - read()で読み込んだ値に対し、制御対象LEDに対応するビットを反転させて、write()で書き込み

実習１：複数のプロセスから同時にLEDを制御 (3/3)

igel

- 動作確認
  1. 1つのLEDを100ms間隔で点滅
  2. 8つのプロセスを同時に起動して、それぞれ異なるLEDを100ms間隔で点滅（複数起動は、シェルの「&」でよい）
  3. 前述動作確認を10msの点滅間隔で10～20回実行
  4. ユーザプログラムにおいて、write()後に設定値を記録しておき、次read()時に読み込んだ設定値と比較、制御対象LEDの（ビット）状態が変わっていたときに警告表示をするコードを追加
  5. 前述の改造を施したユーザプログラムを使って、3. の動作確認を再実行