

演算機能回路

第13週

2006/12/25

泉知論

立命館大学 理工学部 電子情報デザイン学科

モジュール間インターフェース

interfaces

モジュールの種類と入出力

- 組合せ回路モジュール

例：加算器、加減算器、マルチプレクサ

- 入力を与えたそのサイクル内に出力が現れる

- 順序回路モジュール

例：マルチサイクル乗算器、パイプライン乗算器

- 入力を与えてから出力が現れるまで数サイクルかかる

- かかるサイクル数...固定？不定？

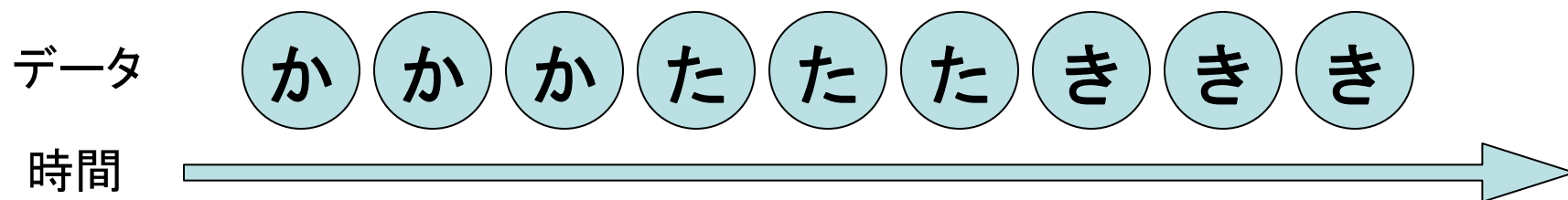
- 出力をいつ受取るのか？

- そもそも入力をいつ渡すのか？

疑問

- ソフトウェアの関数
 - 呼び出されたときに入力を受取る
 - 戻るときに出力を返す
- ハードウェアのモジュール
 - 常に存在して、動いている（待機状態かもしれないが）
 - 信号線には常に何らかの値（電圧）がのっている（値が無いということは有り得ない）
 - いつ値を受取る？
 - いつ値を返す？

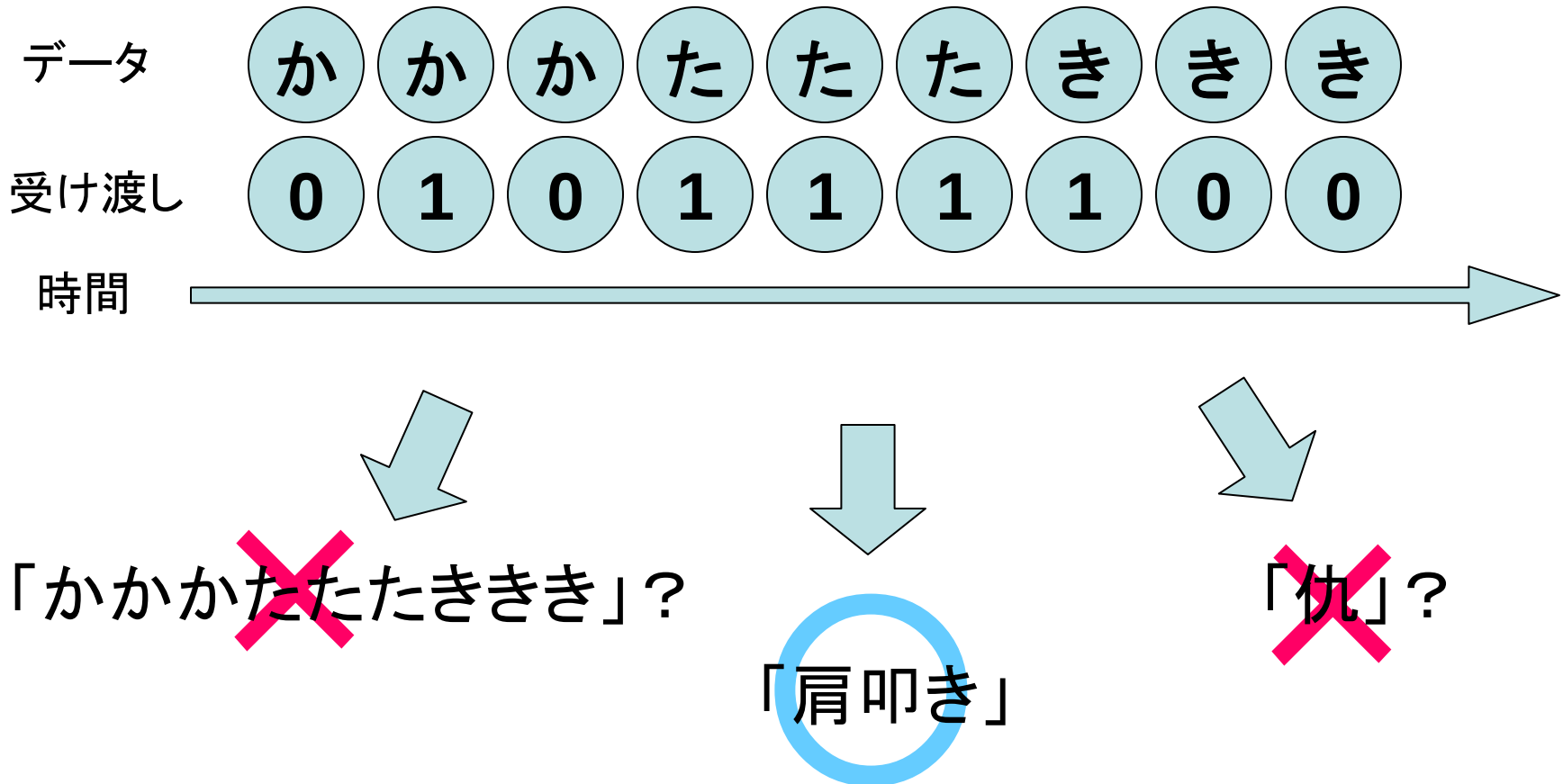
信号を受け取るタイミングが不明



「かかかたたたたききき」?

「仇」?

信号を受け取るタイミングを指示



タイミング指示の方法

- 固定型
 - 送受のタイミングを事前に固定
 - ステートマシンが重複
 - 慎重なタイミングの摺り合わせが必要
 - メンテナンス性が低い(デバグ、改造、使いまわし)
- 集中管理型
 - ひとつのコントローラが各モジュールに指示を出す
 - 回路はシンプル
 - 結局、コントローラがパンドラの箱！？
- 自律型
 - 送受モジュール同士でタイミングを決める
 - 制御のオーバーヘッド

信号送受の主導権

送る側



送りたい

送れない

受ける側



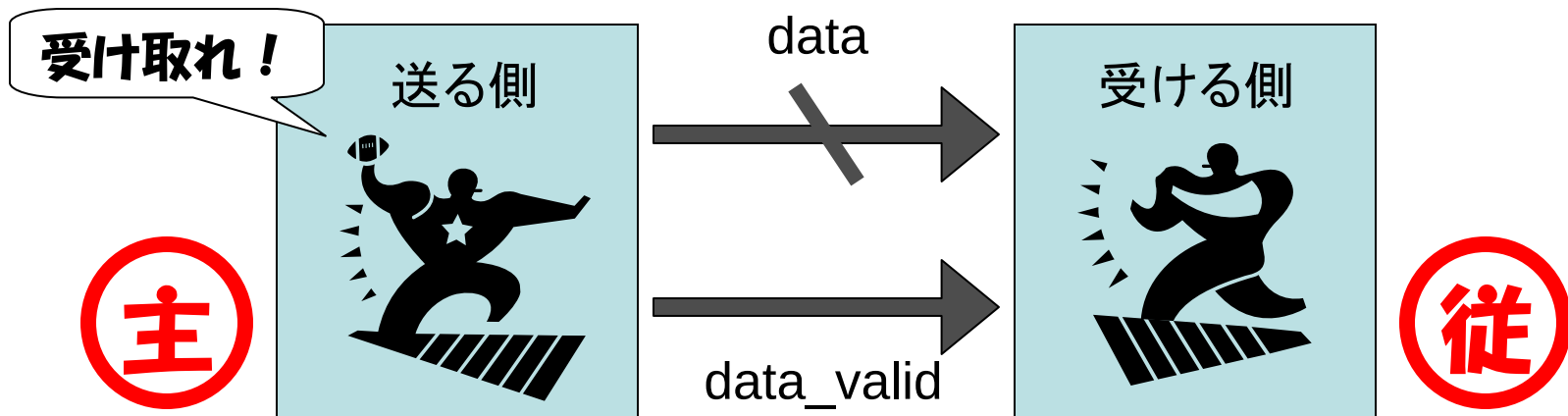
受けたい

受けられない

- 送る側が主導
 - push 型
- 受ける側が主導
 - pull 型
- 対等に交渉
 - push&pull 型
 - handshake 型
- これらの組合せ

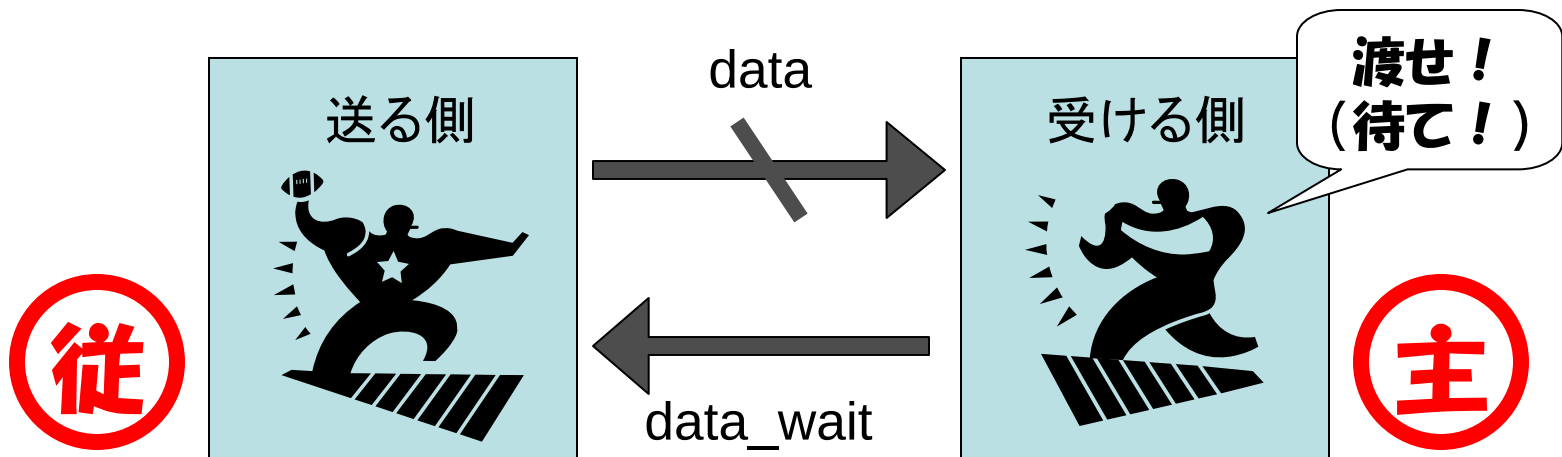
push型

- 送る側が送りたいときに送る
 - 受ける側はいつでも受信可能状態でなければならない
- データ線にデータ送信を示す信号線を加える
 - 信号名の例: valid (有効), enable (可能にする), strobe (閃光), out (出力), do (しなさい), など



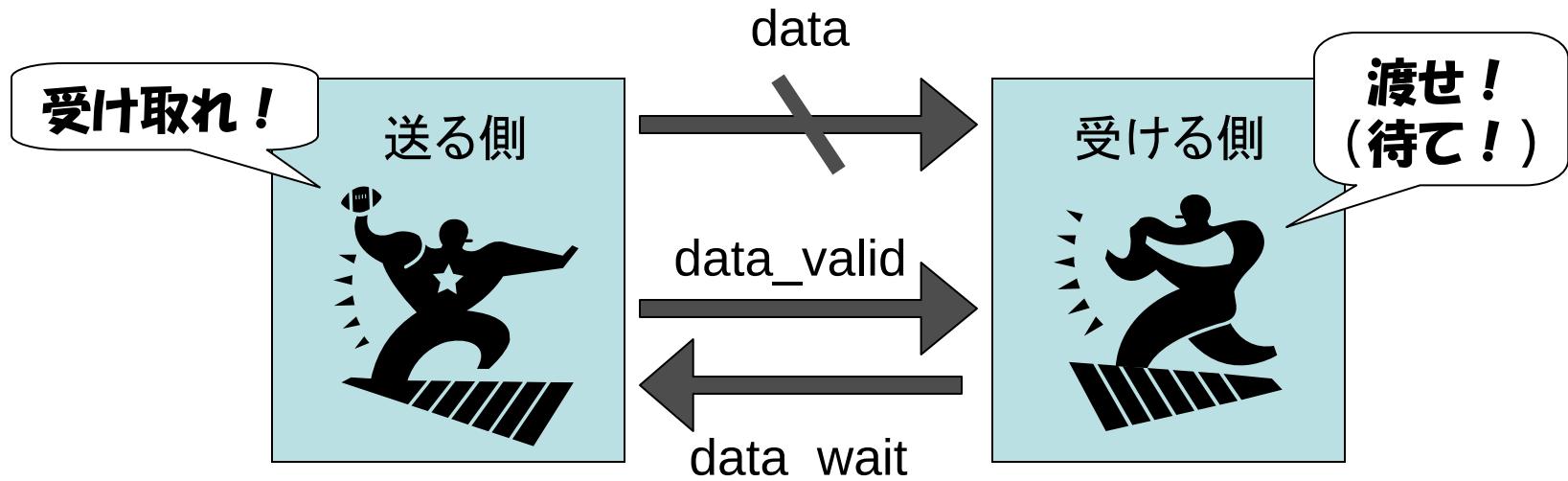
pull型

- 受ける側が受けたいときに受ける
 - 送る側はいつでも送信可能状態でなければならない
- データ線にデータ受信を示す信号線を加える
 - 信号名の例: receive (受取る), wait (待て), in (入力) など
 - 設計の都合上、「今渡せ」と「次に渡せ」のバージョン有



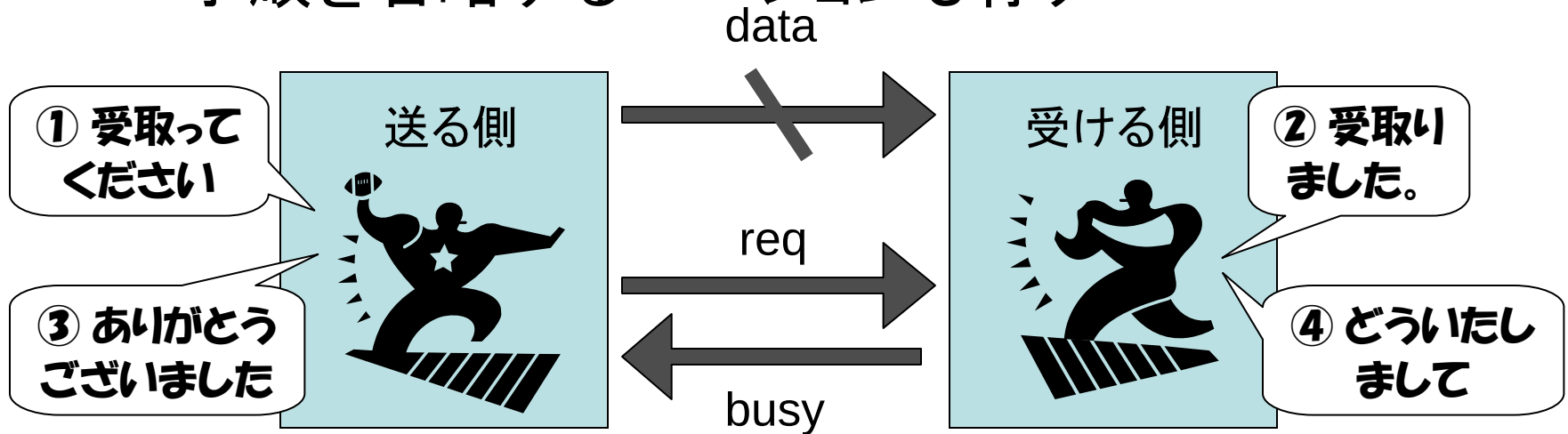
push&pull型

- 送りたいかつ受けたいときに送受
 - その分、制御が複雑になる
- データ線にデータ送受信を示す信号線を加える

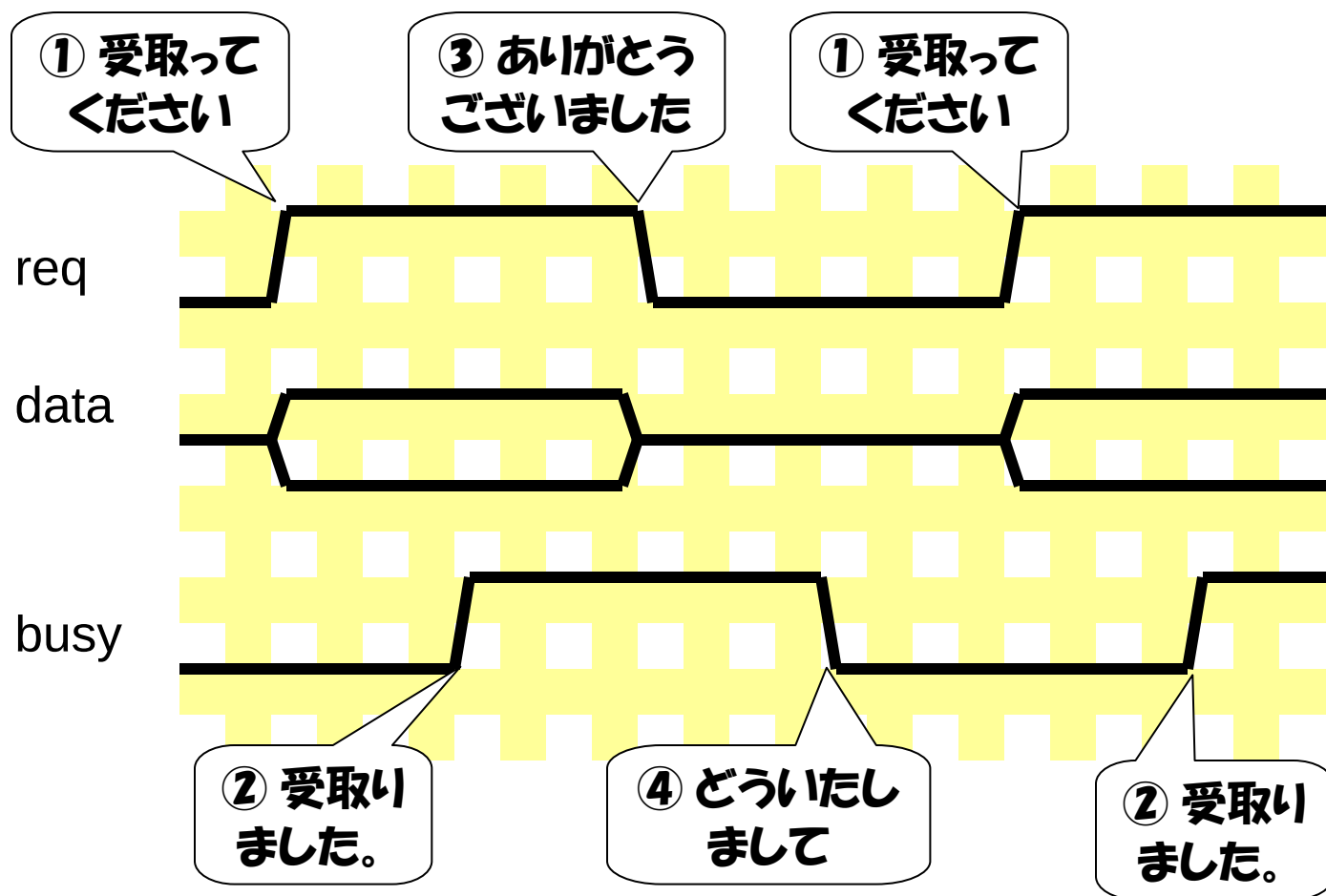


handshake型

- お互いの状況を確認しあいながら順にすすめる
 - 確実だが手間(制御回路、時間)がかかる
- データ線に送受状況を示す信号線を加える
 - request 信号と acknowledge/busy 信号の2線4相
 - 手順を省略するバージョンも有り

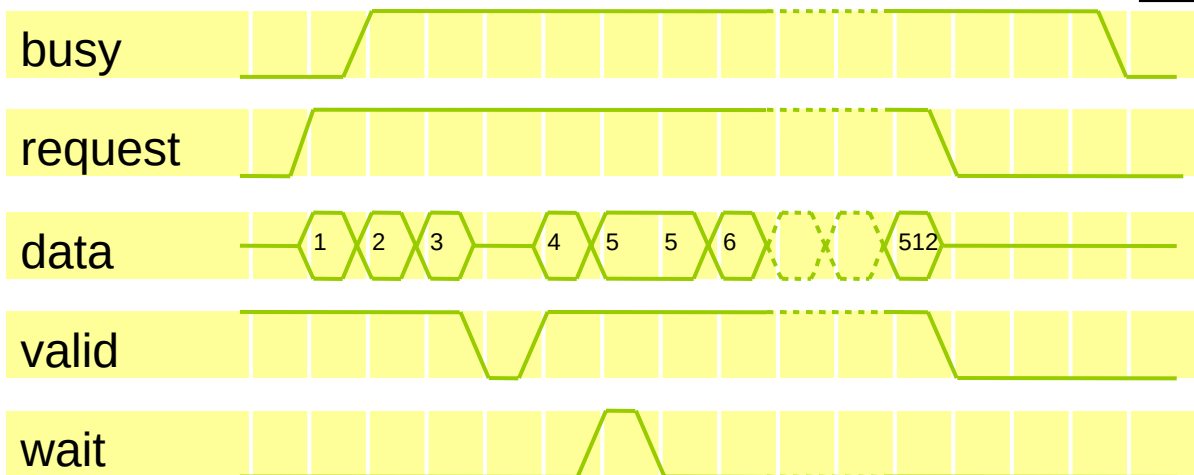
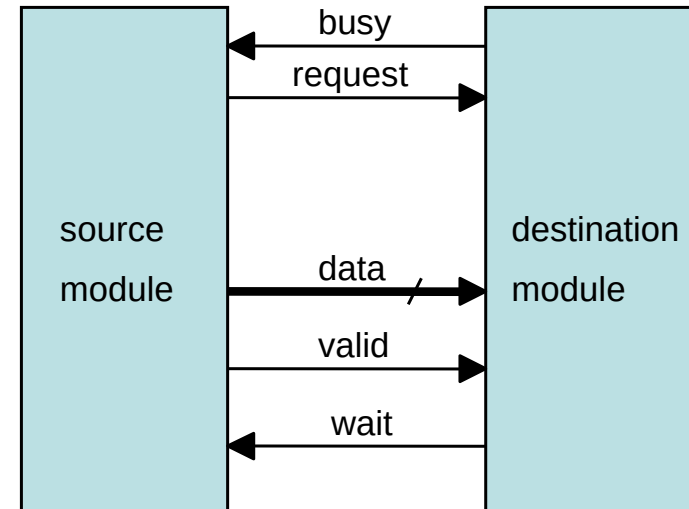


handshake 型(続き)



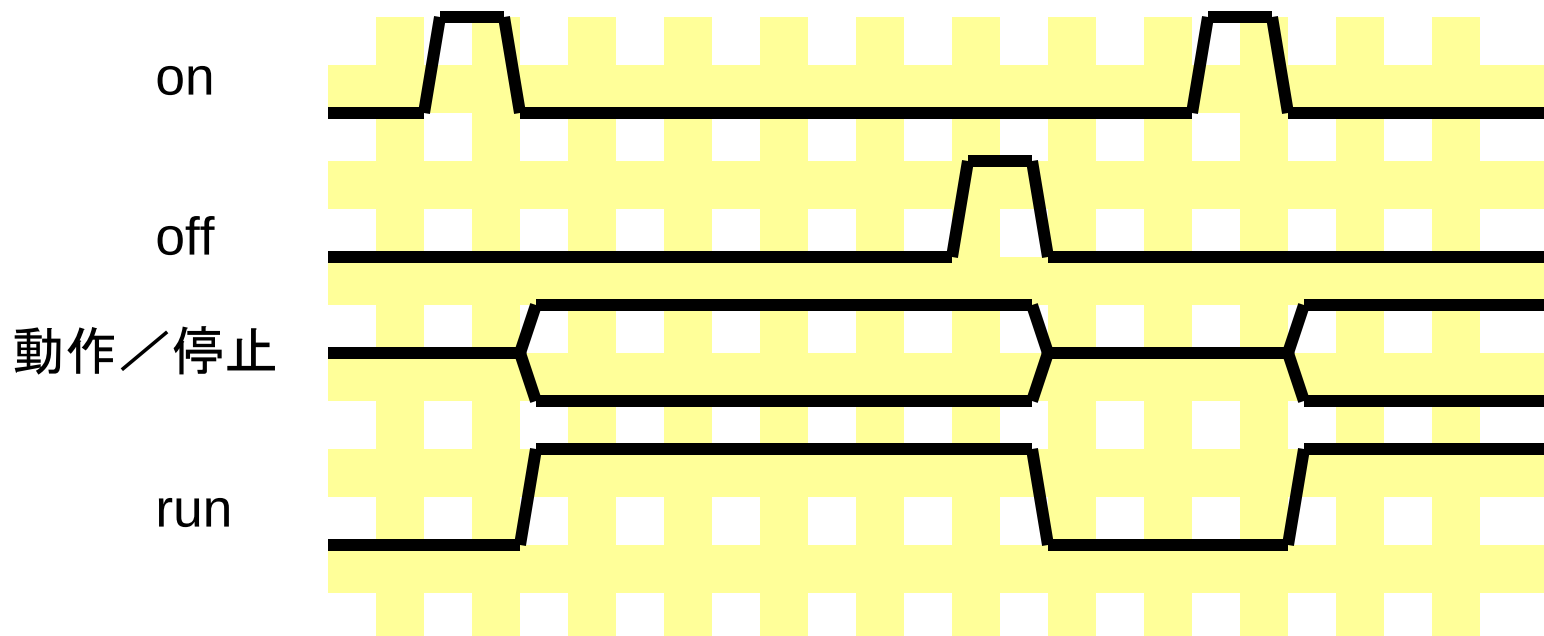
複合型モジュール間I/Fの例

- 画面単位の手shake 制御
- 画素単位の手shake&pull 制御



パルス信号とレベル信号

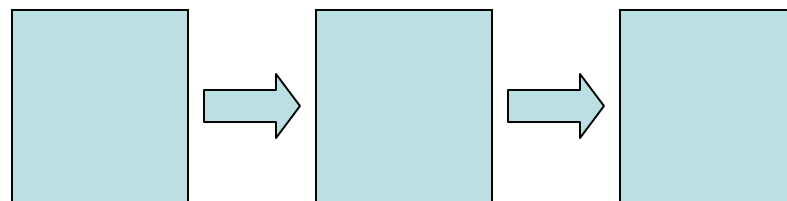
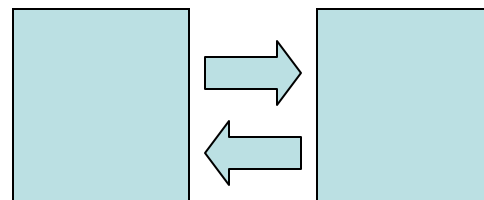
- パルス信号
 - 1サイクルのパルスで意図を示す
- レベル信号
 - 0か1かで意図を示す



接続形態

- 専用接続

- 往復型
- 一方通行型



- バス接続

- バスコントローラ(アービタ、DMA、割り込み)

