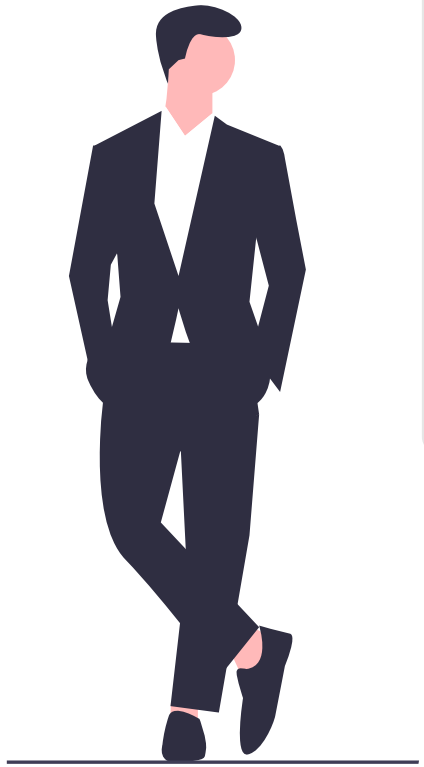
ハーバードマシン | 事前報告

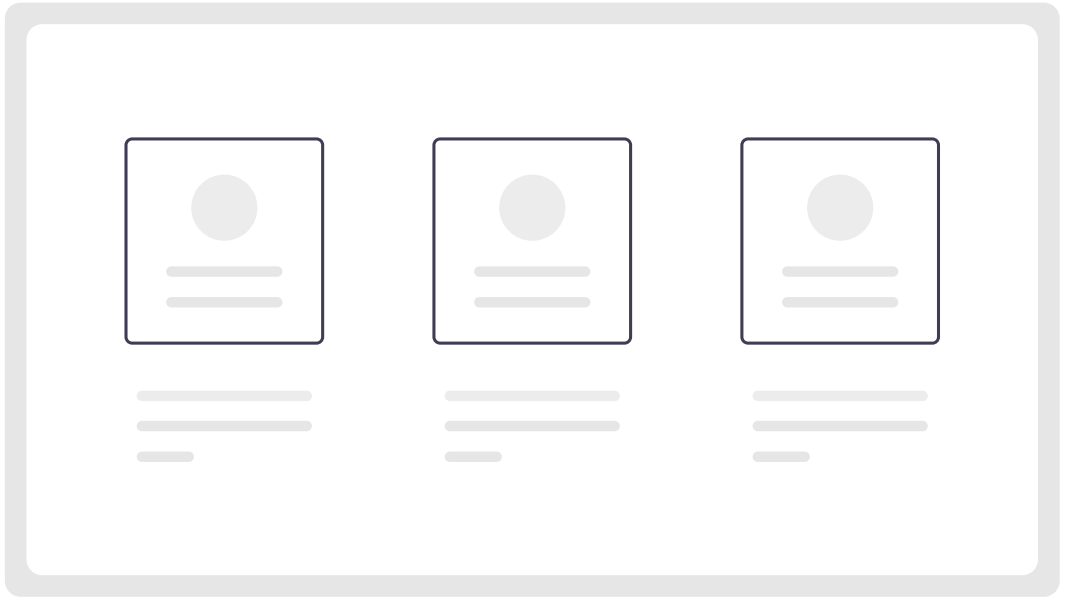
発表　　■■ ■■　■■ ■■　■■ ■■

発表資料　　■■ ■■

01

About





機能

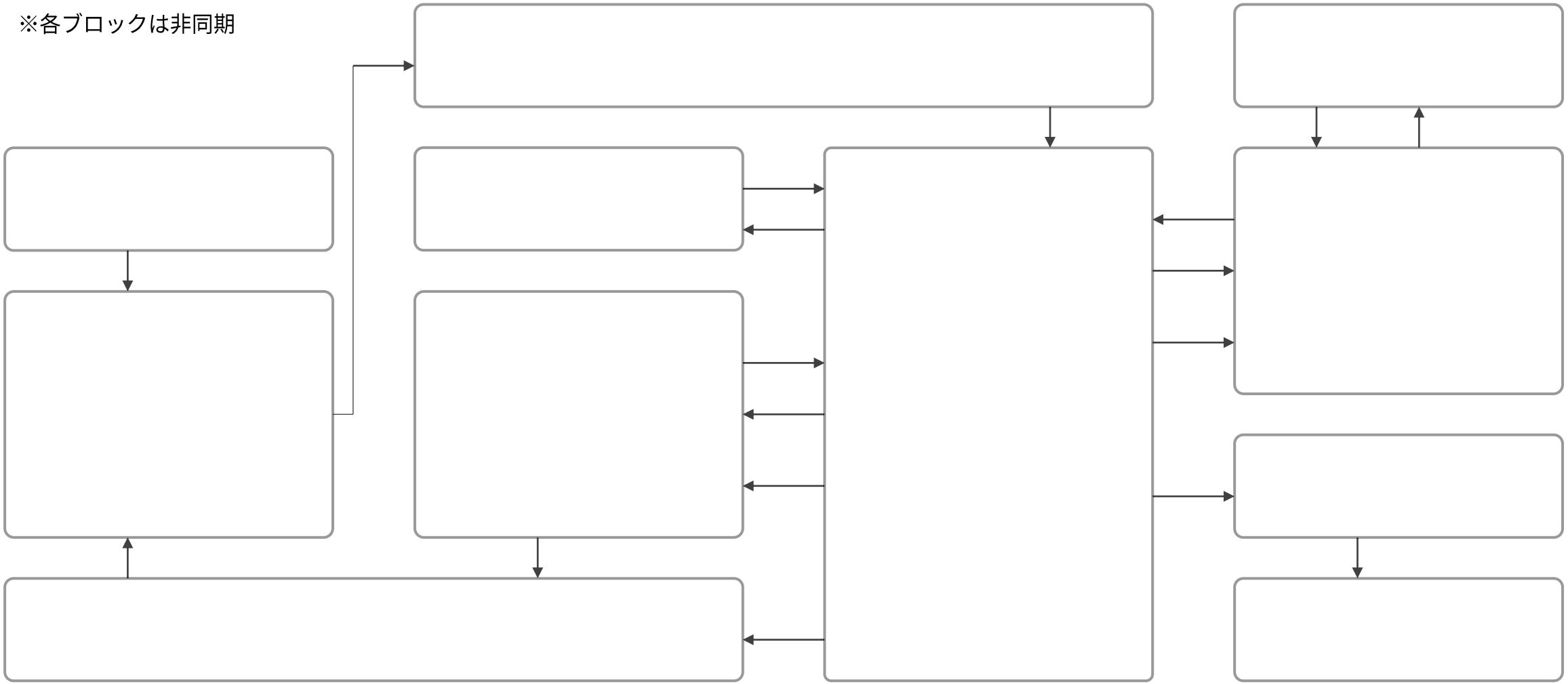
・あらかじめ書き込まれた命令に従って算術･論理演算を行える
・上記で，演算した結果を7セグメントLEDを使って表示できる
・高級言語で記述したソースコードをPC上でアセンブルできる

利点

・4年「計算機ハードウェア」で学習した知識をより深められる

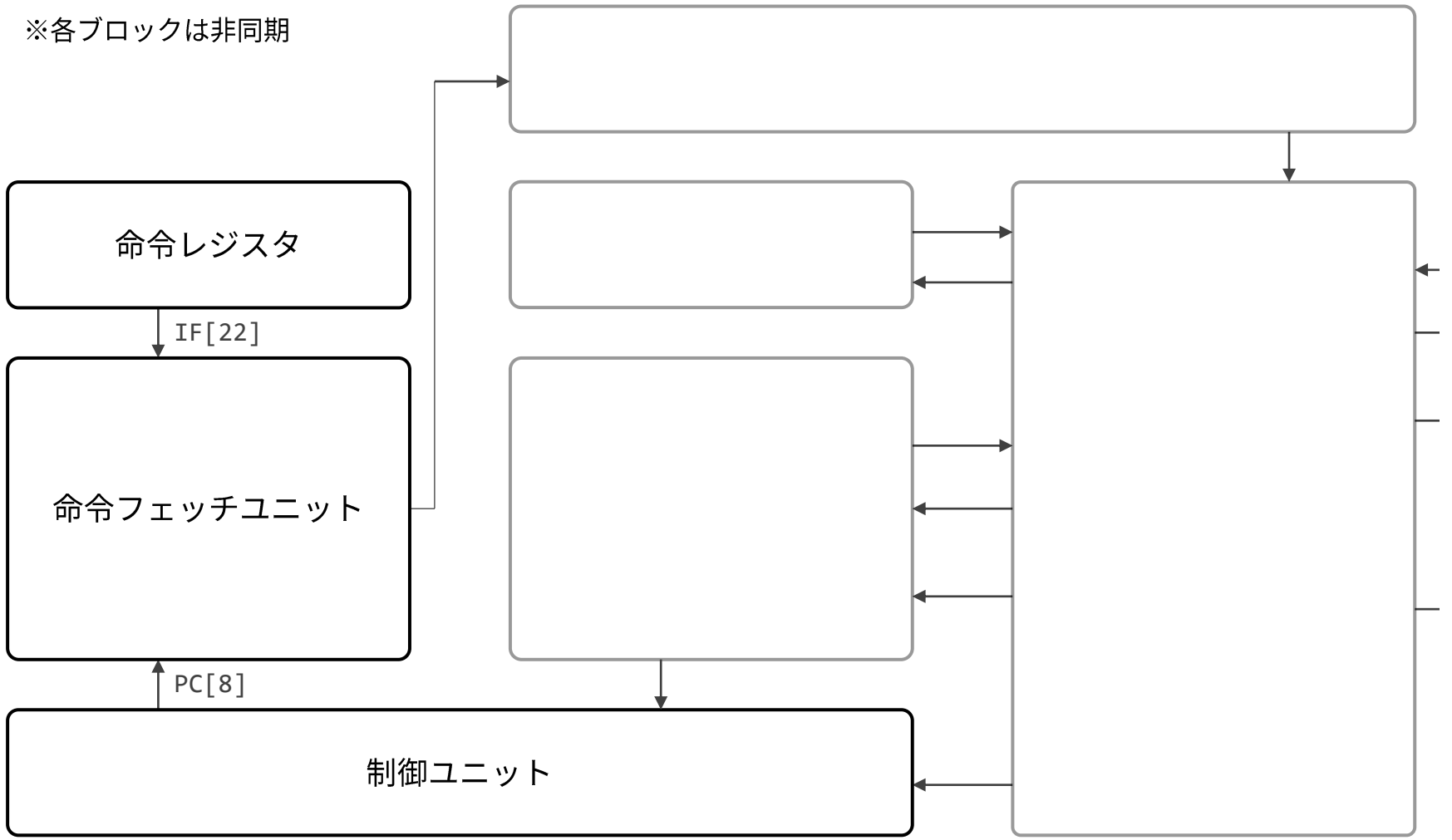
機能ブロック図 | Function Block Diagram

※各ブロックは非同期



機能ブロック図 | Function Block Diagram

※各ブロックは非同期



命令レジスタ

命令を保管する

命令フェッチユニット

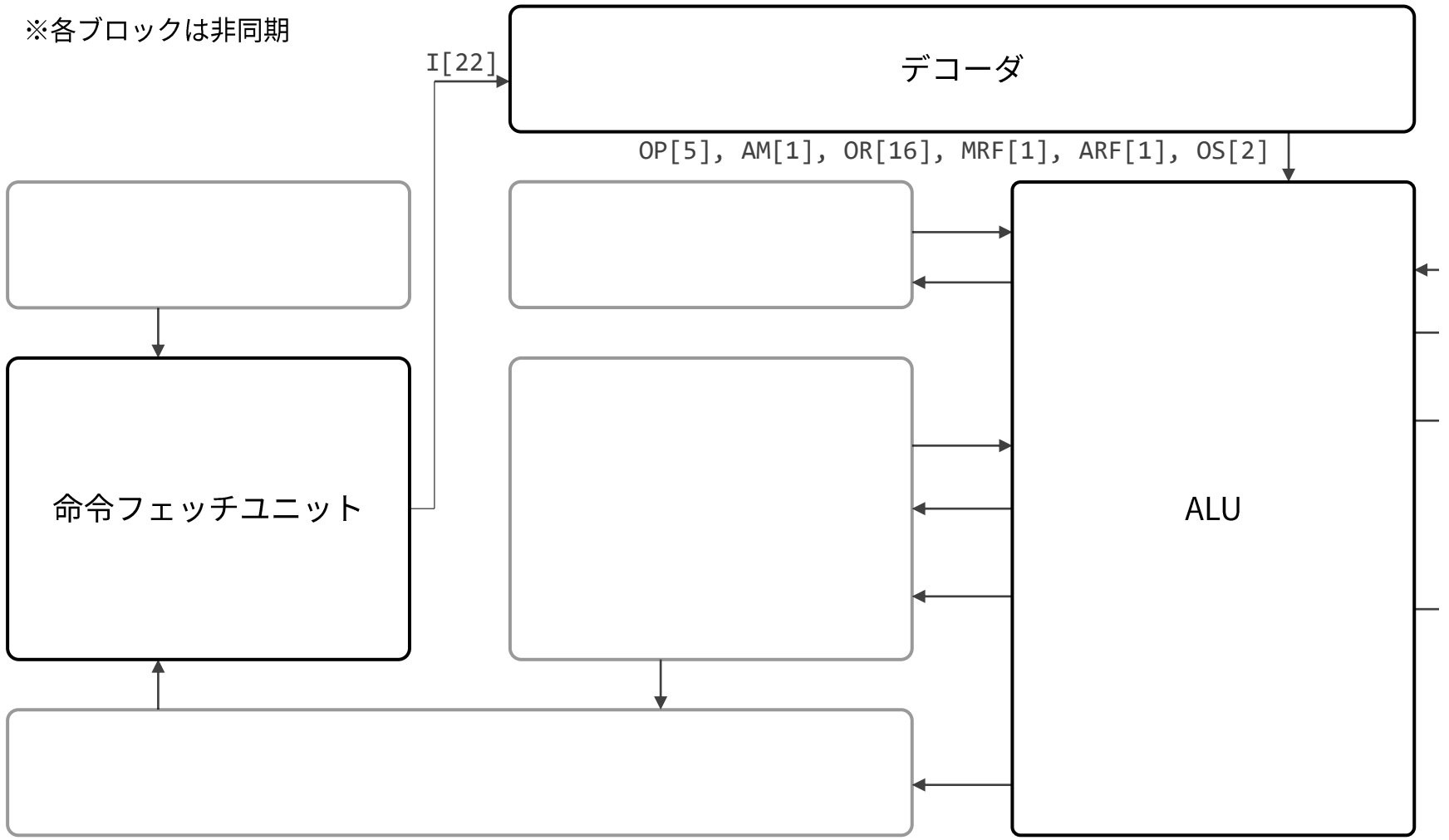
命令を取り出す

制御ユニット

取り出すアドレス
＝PCを指定する

機能ブロック図 | Function Block Diagram

※各ブロックは非同期



デコーダ

命令を読み込む

ALU

様々な計算を行う

機能ブロック図 | Function Block Diagram

累積器

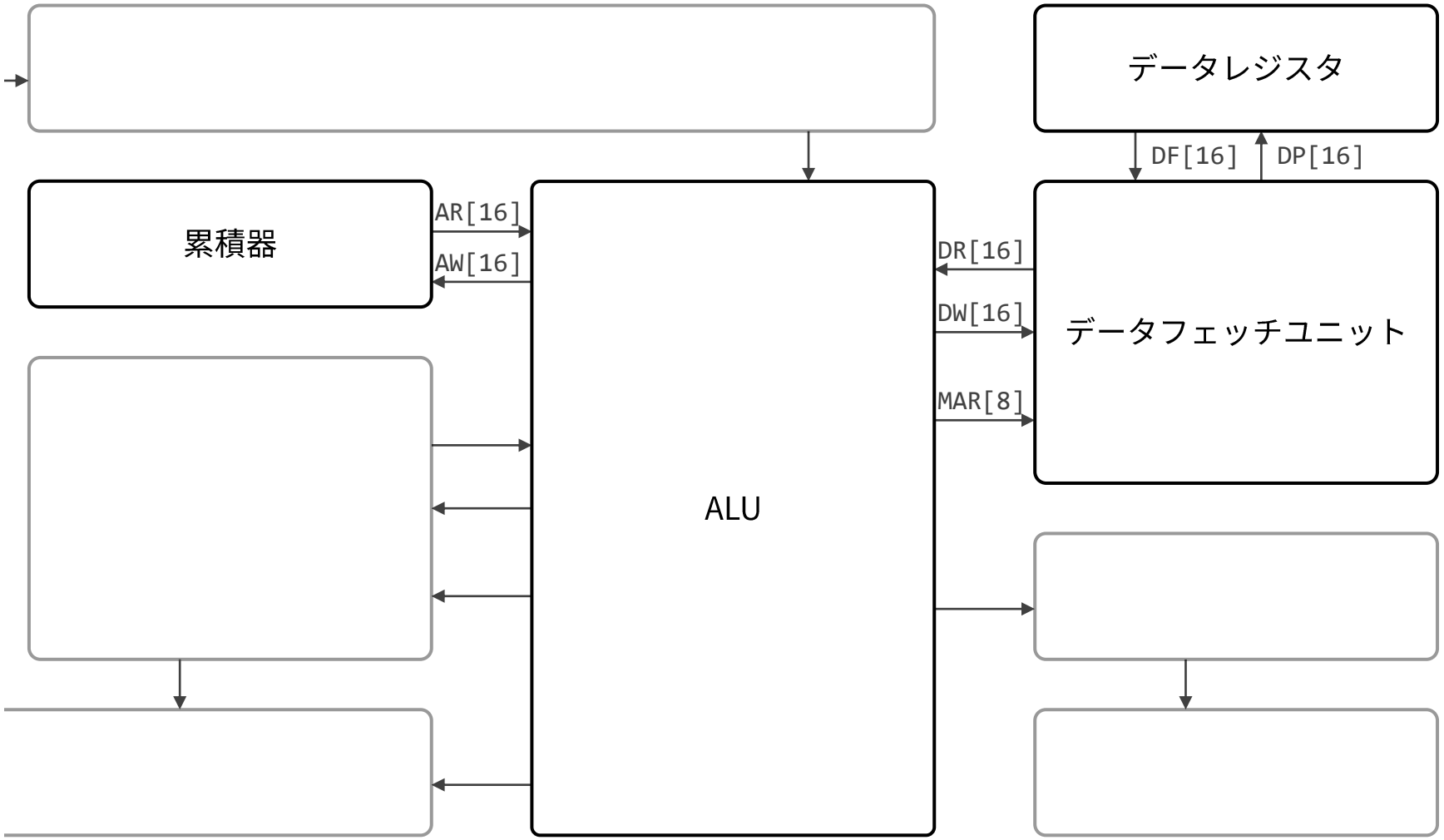
データを一時的に記憶する

データレジスタ

データを保管する

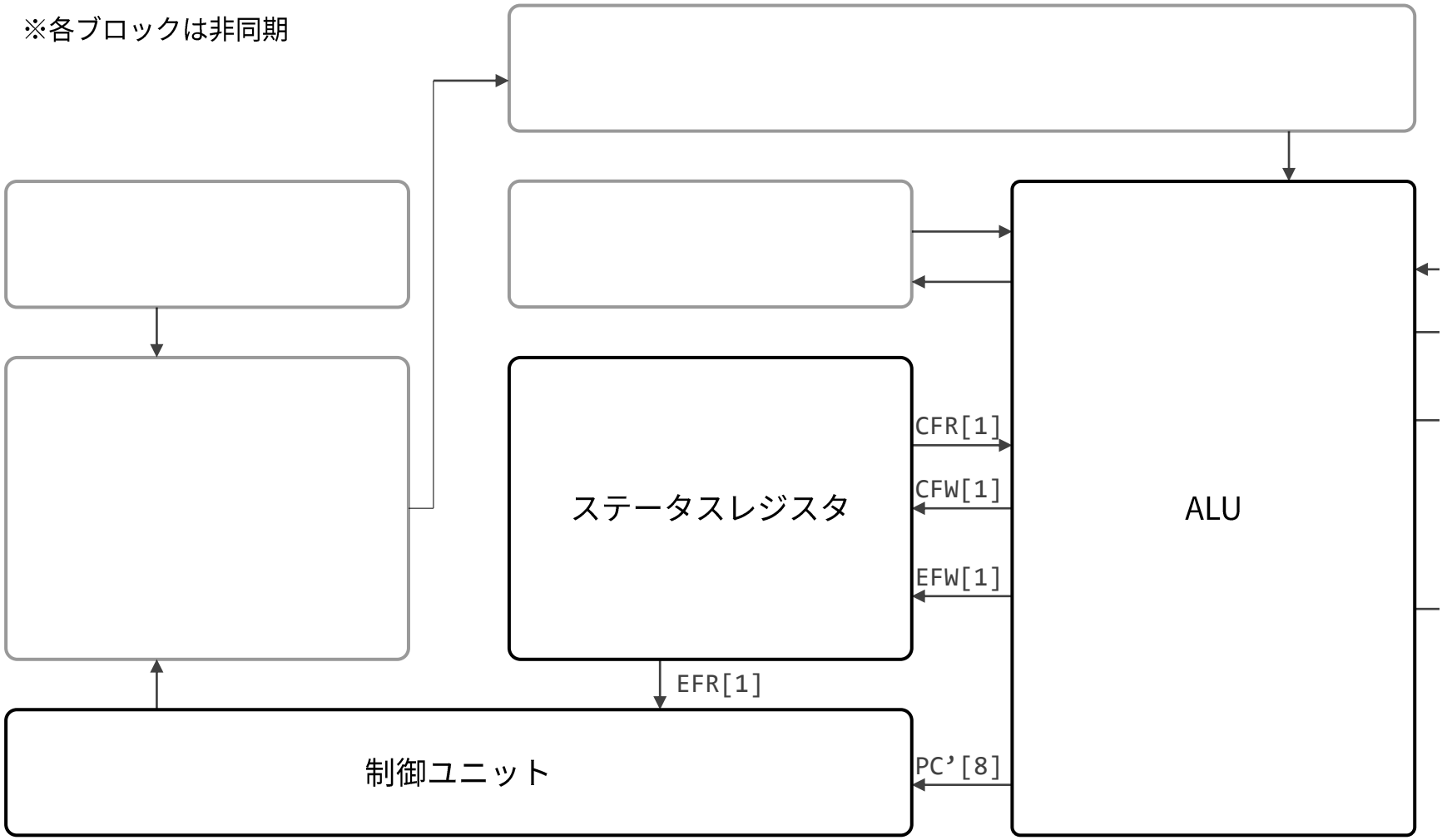
データフェッチユニット

データを取り出す



機能ブロック図 | Function Block Diagram

※各ブロックは非同期



ステータスレジスタ

演算に付随する情報を保管する

制御ユニット

取り出すアドレス＝PCを指定する

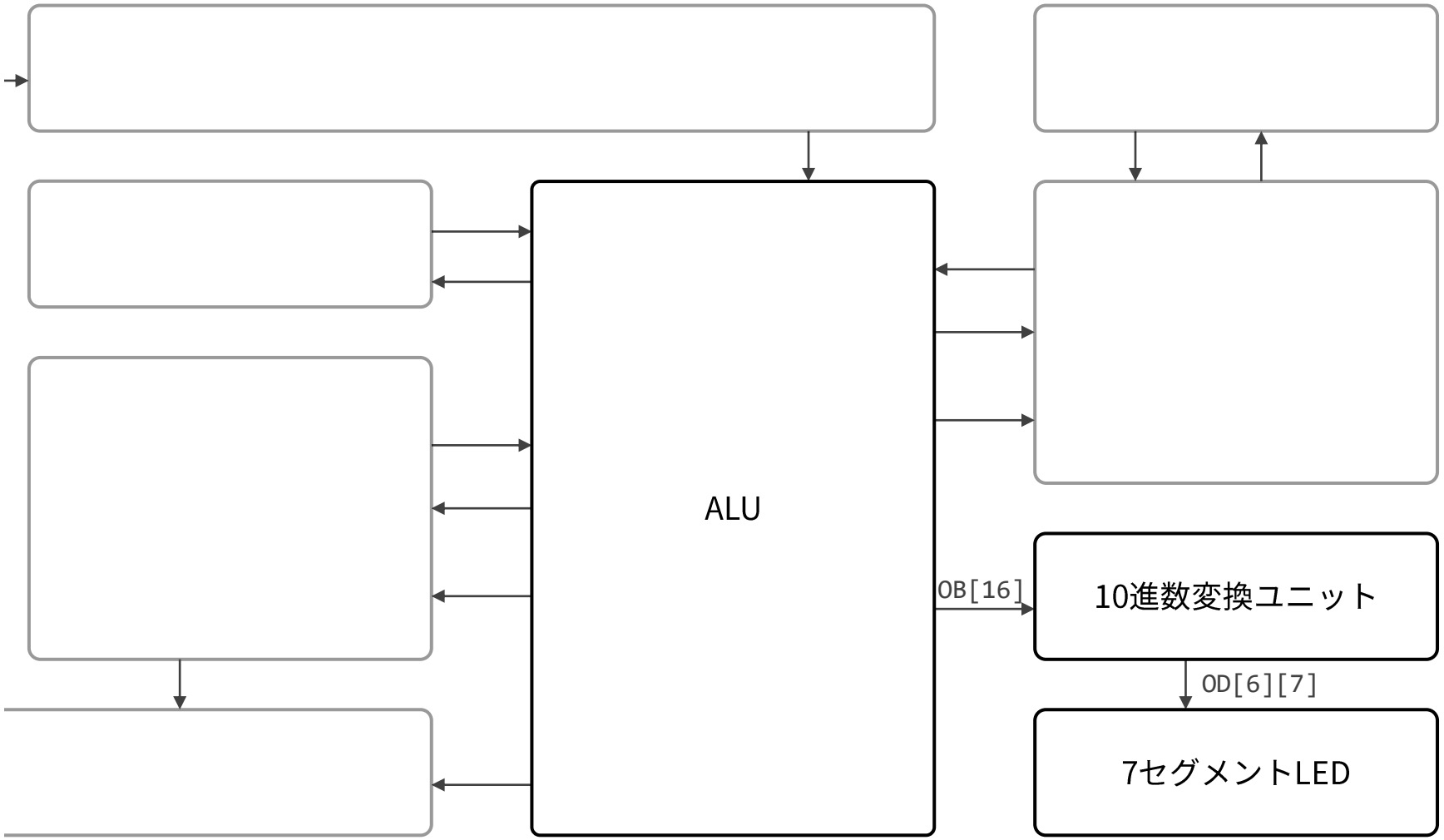
機能ブロック図 | Function Block Diagram

10進数変換ユニット

データを10進数に
変換する

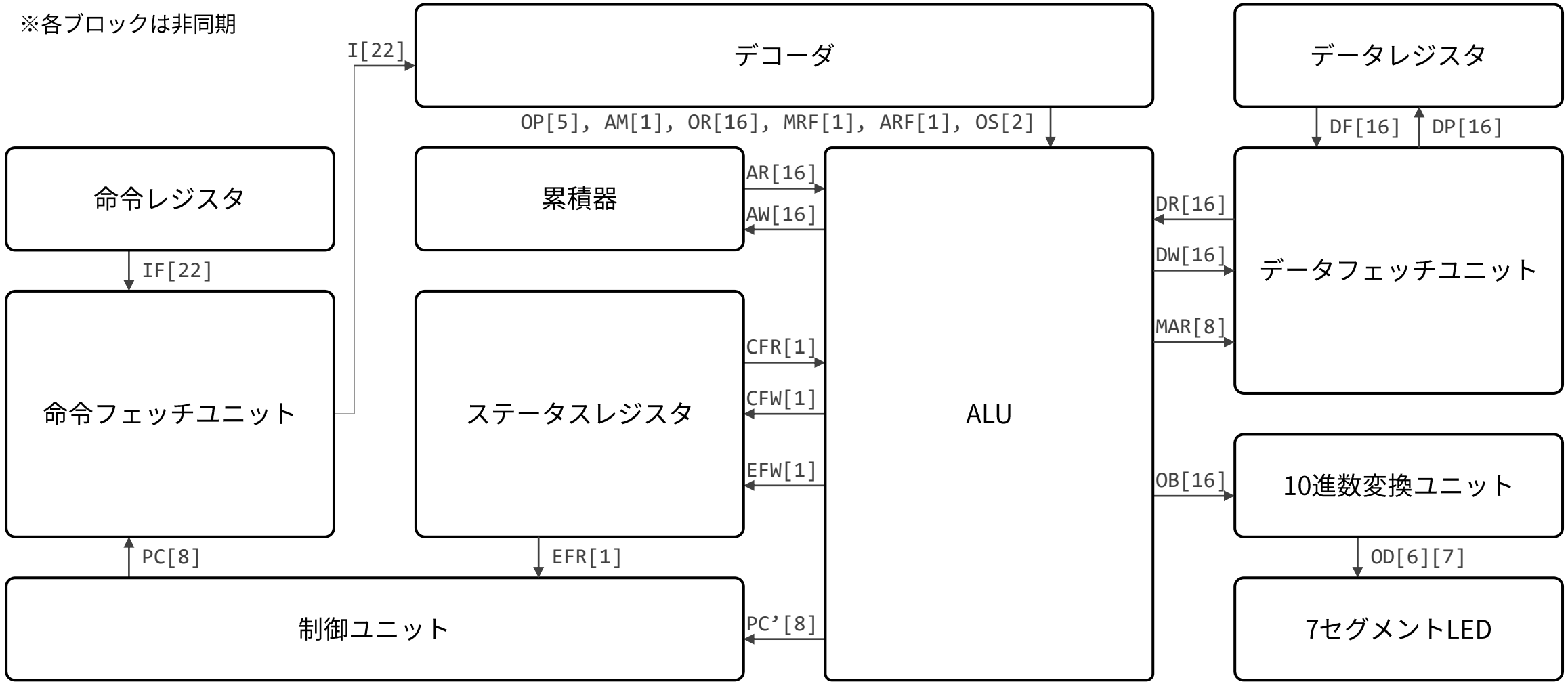
7セグメントLED

符号･数字を表示する



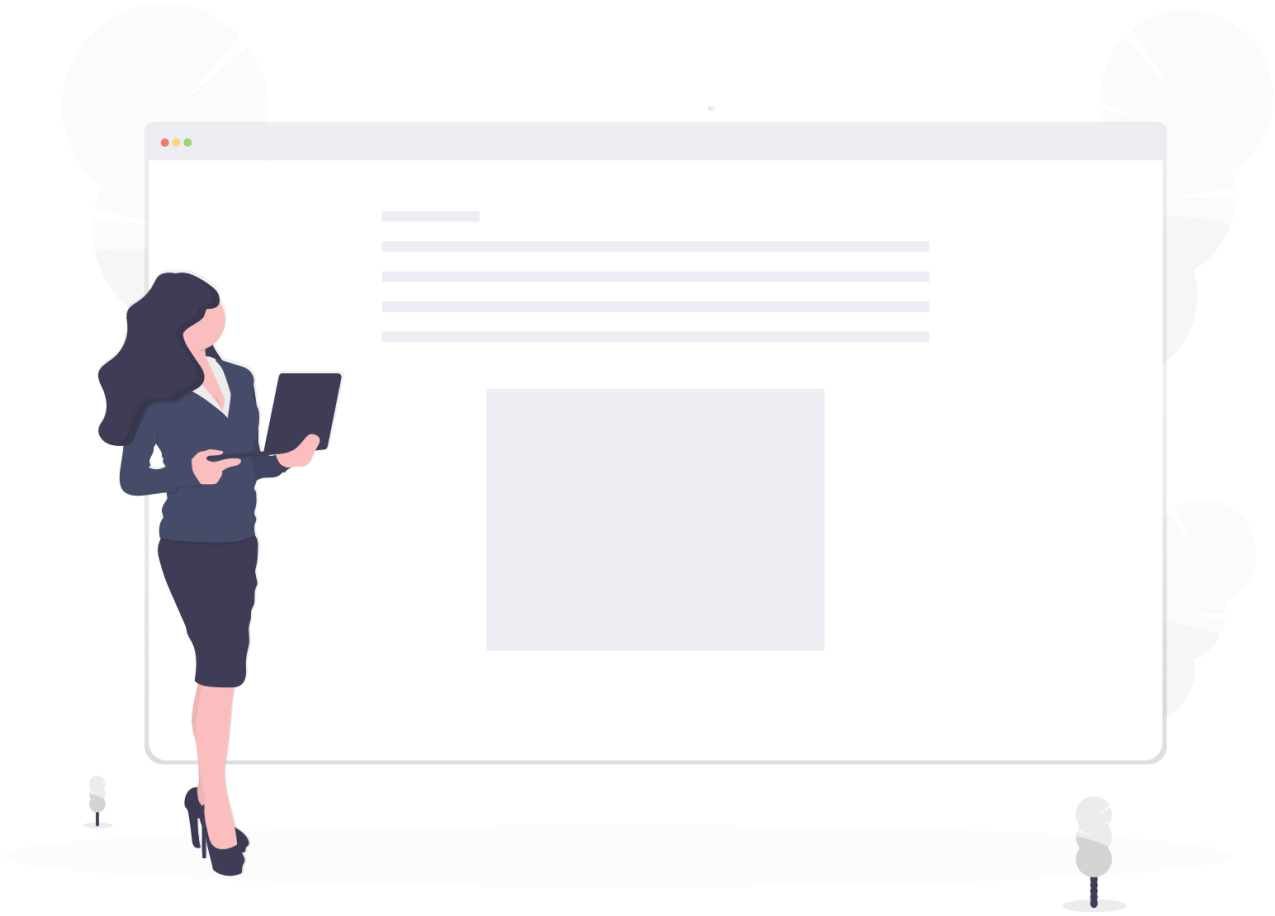
機能ブロック図 | Function Block Diagram

※各ブロックは非同期



02

Detail



アドレス

アドレスを表す値（PC，MAR）は，ともに8bit

データ

データは全て16bit整数型（32767 ～ -32768）
ただし，アドレス値を表すデータは下位8bitのみ使用される

```
0000 0000 0101 1010
          アドレス表現に使用される部分
```

命令

各命令は22bit固定長（オペコード5bit，アドレス指定モード1bit，オペランド16bit）

オペコード	アドレス指定方式	オペランド
ADD	IMM	0000 1111 0011 1100

命令長を固定するため，実際には使用されないオフセットを含むことがある

例：引数を取らない制御命令は，命令長を22bitにするため適当な値でその部分を補間する

オペコード	アドレス指定方式	オペランド
NOP	0	0000 0000 0000 0000

命令セット | Instruction Set

命令	引数	説明
ADD	あり	累積器の値に引数を加算し，結果を累積器にセットする
SUB	あり	累積器の値に引数を減算し，結果を累積器にセットする
MUL	あり	累積器の値に引数を乗算し，結果を累積器にセットする
DIV	あり	累積器の値に引数を除算し，結果を累積器にセットする
MOD	あり	累積器の値に引数を余算し，結果を累積器にセットする
NOT	なし	累積器の値の論理否定をとり，結果を累積器にセットする
OR	あり	累積器の値と引数の論理和をとり，結果を累積器にセットする
AND	あり	累積器の値と引数の論理積をとり，結果を累積器にセットする

命令セット | Instruction Set

命令	引数	説明
LOAD	あり	引数を累積器にセットする
STORE	あり	累積器の値を引数のアドレスにセットする
LT	あり	累積器の値が引数より小さい時0，それ以外の時1をCFにセットする
GT	あり	累積器の値が引数より大きい時0，それ以外の時1をCFにセットする
BEQ	あり	CFが0の時，PCの値を引数に変更する
BNE	あり	CFが1の時，PCの値を引数に変更する
JUMP	あり	CFによらず，PCの値を引数に変更する
PRINT	あり	引数を出力する
NOP	なし	何もしない
EOP	なし	プログラムの終了を示す

アドレス指定方式 | Addressing Mode

方式	説明
ABS	〈絶対アドレス指定モード〉 オペランドをアドレスと解釈し，参照先のデータを引数とする
IMM	〈即値指定モード〉 オペランドをデータと解釈し，そのまま引数とする

高水準言語

C言語をベースとした手続き型言語で，「+」「-」「*」「/」等の
演算子を用いて四則演算・論理演算ができる

実装予定

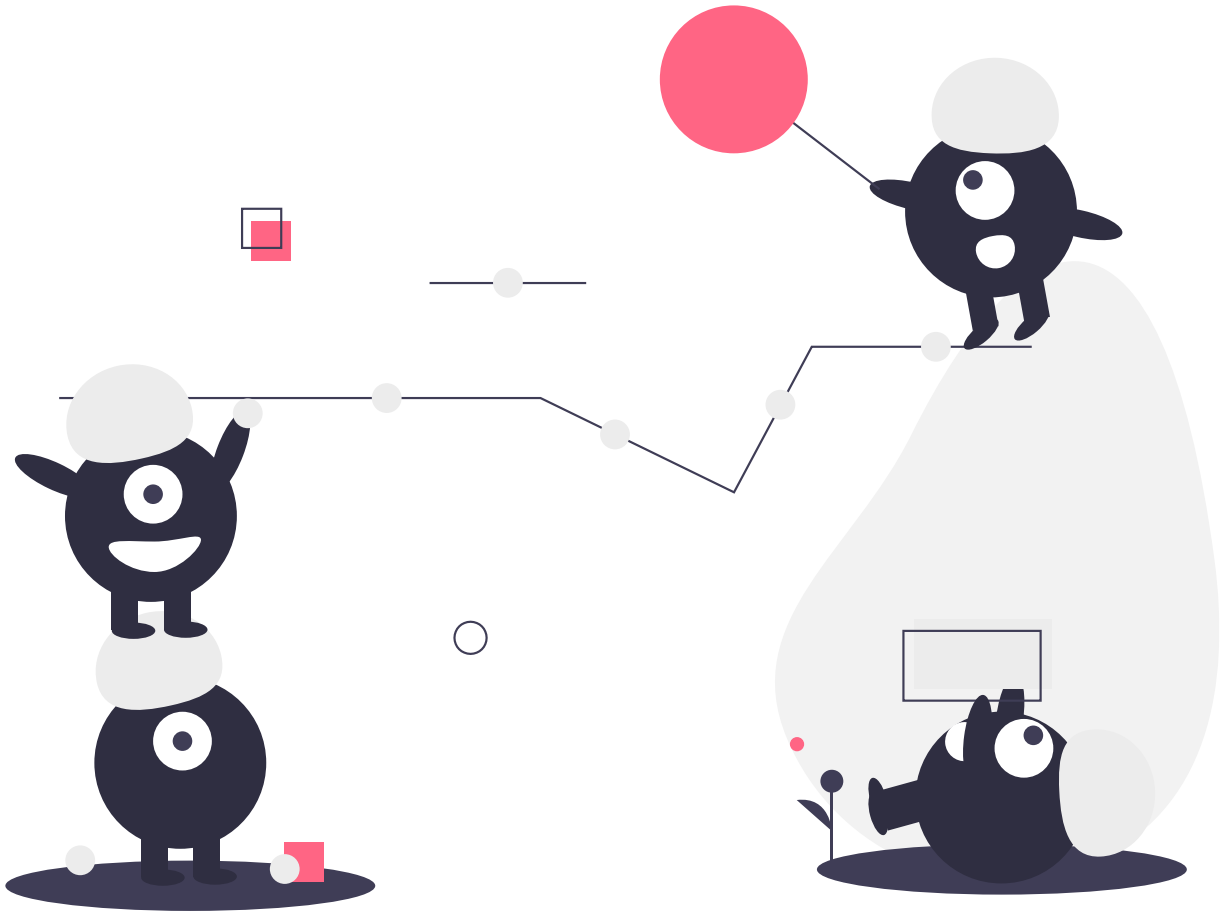
変数・定数	関数	if文
while文	print	sleep

実装見送り

配列	ポインタ	構造体
for文	switch文	

03

Team



メンバー | Member

■■ ■■ CEO

命令セット仕様　デコーダ設計･実装

マネジメント　スケジュール管理

■■ ■■ CTO

レジスタ&フェッチユニット設計･実装

制御ユニット設計･実装

■■ ■■ エンジニア

■■･■■付き実装補助　プログラム

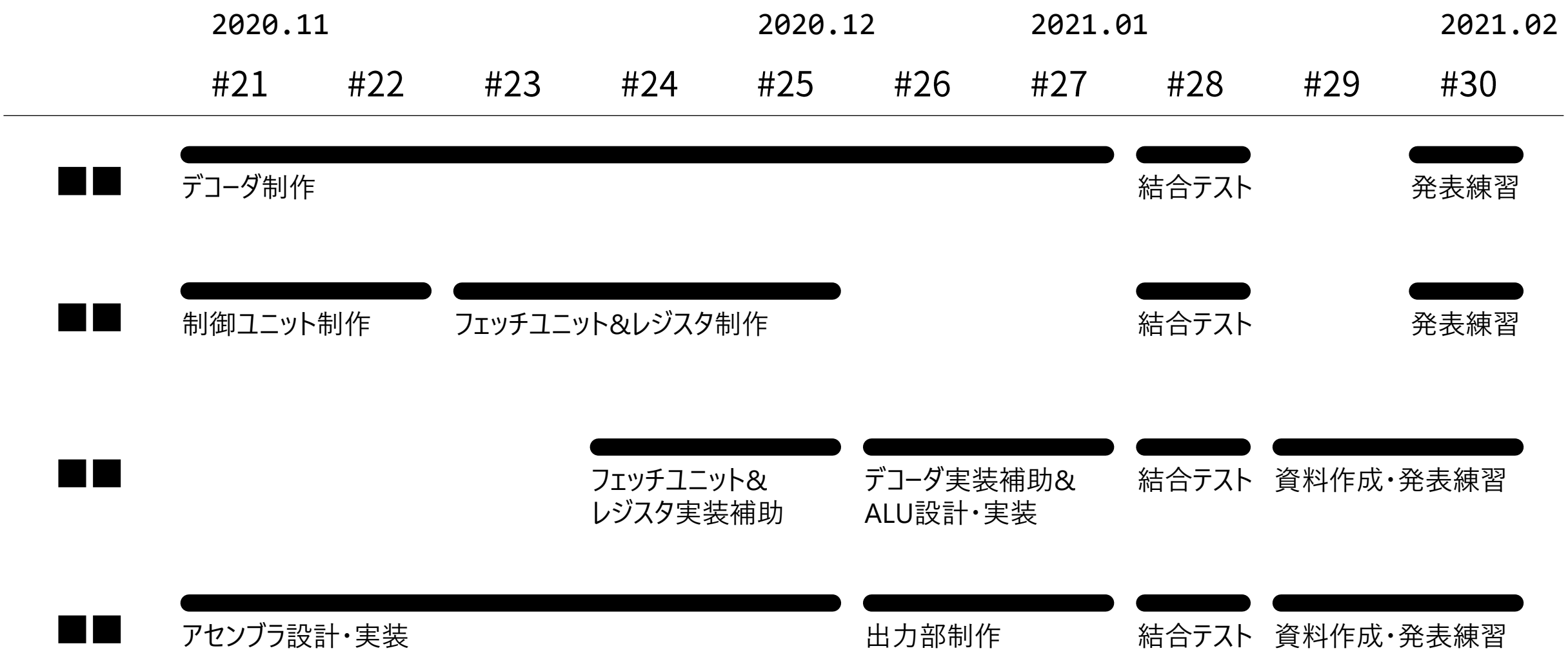
ALU&累積器設計･実装　資料作成

■■ ■■ デザイナ

出力部設計･実装　高水準言語仕様

アセンブラ設計･実装　資料作成

スケジュール | Schedule



開発環境 | Development Environment

使用機器	FPGA：DE10-Lite
購入物品	—
開発環境	Intel Quartus Prime（FPGA）
開発言語	Verilog HDL（FPGA）　Python（アセンブラ）
VCS	GitHub