


CISCの極み Intel iAPX 432の紹介

第5回 自作CPUを語る会 2025/04/12

@uchan_nos



-  iAPX 432 とは
- システム構成と Multibus
- 2 階層のアドレス変換
- 機械語の構造
- オブジェクト保護

- 何かのきっかけでWikipediaの記事を読んだ
 - https://ja.wikipedia.org/wiki/Intel_iAPX_432
- x86に親しんだ身からすると、かなり奇抜な設計に思えた
- BuntanPCプロジェクトの参考になりそうだと思い、調査開始
 - 「コンピュータ再設計プロジェクト」
 - 積極的に「変な設計」を試してみたい

- Intel Advanced Processor architecture
- 1981 年に発表
- 32 ビット、8MHz
 - 2 年前に発表された MC68000 も 32 ビット 8MHz
 - 1982 年に発表された Intel 80286 は 16 ビット 8MHz
 - 1985 年に発表された Intel 80386 は 32 ビット 12MHz (?)
- メモリ管理やマルチタスクをハードウェアサポート
- 3 チップ構成
 - 43201: 命令のフェッチとデコード
 - 43202: マイクロコードの実行とメモリアドレス生成
 - 43203: iAPX 432 と I/O デバイス間の通信とデータ転送


- スタックマシン型の命令実行
 - RAM にオペランドスタックを配置
 - 汎用レジスタは無い
- 命令長はビット単位の可変長
 - 1 命令の長さは 6 ビット～300 ビット超
 - 命令セグメントは 16 ビットアドレス = 8KB が上限
- 徹底した間接アドレッシング
 - プログラマが生のアドレスを指定することが無い（指定できない）
 - すべてのメモリアクセスはアクセス記述子を使う
- 非常に高度な命令がある
 - プロセス間でメッセージを送受信するための命令（SEND/RECEIVE）
 - 型定義に基づきインスタンスを生成する命令（CREATE_TYPED_OBJ）

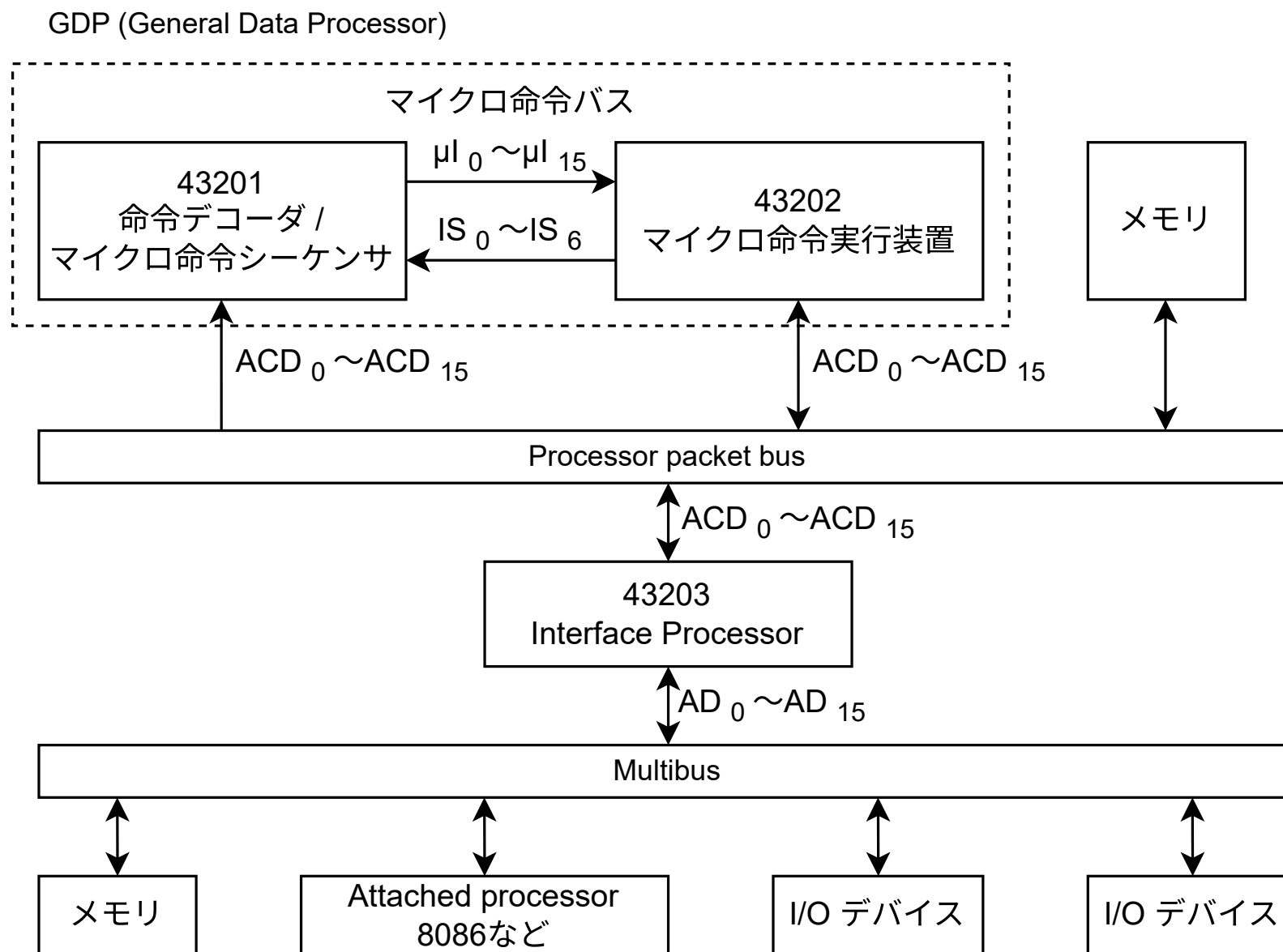
実行時間の比較（単位はミリ秒）

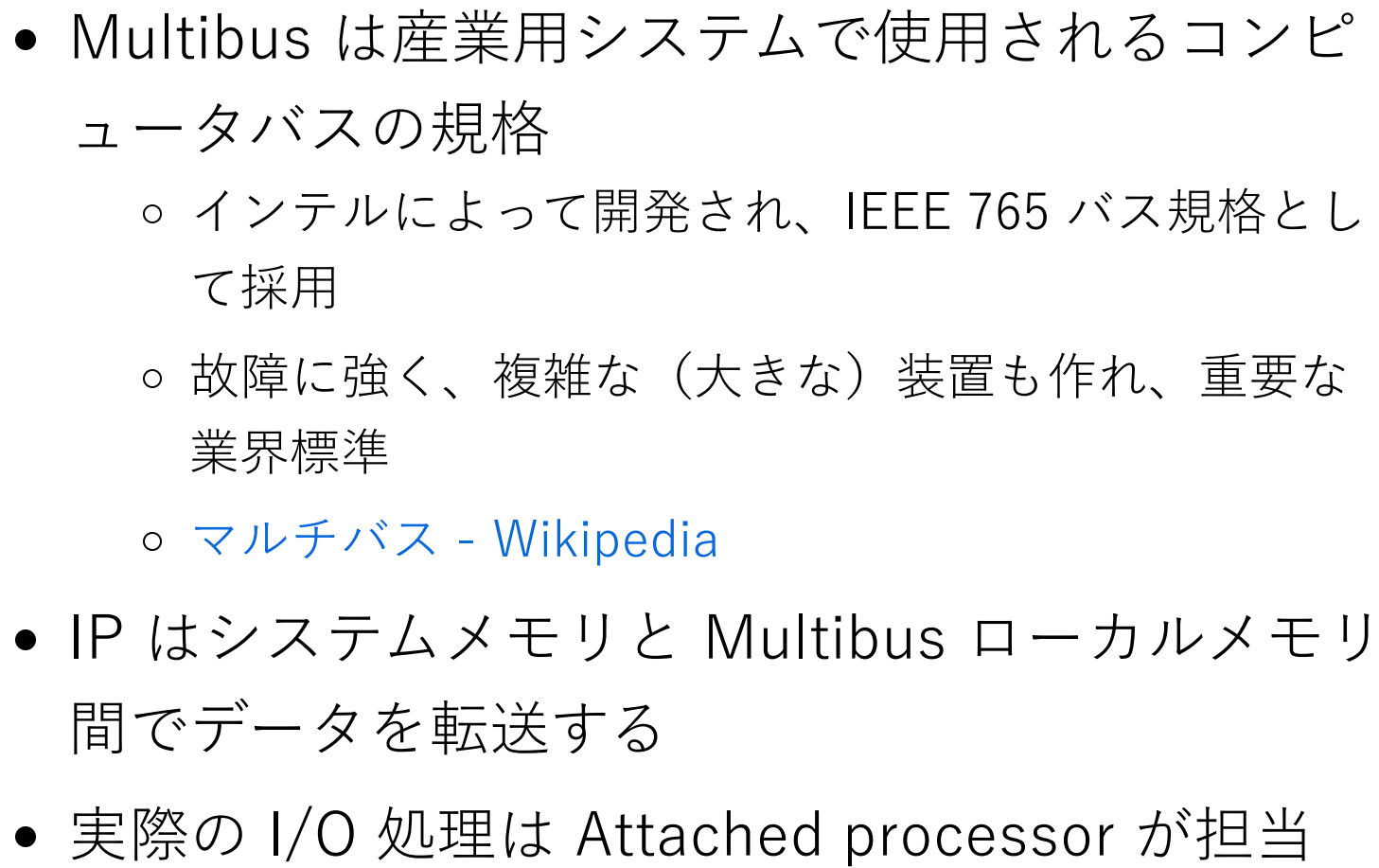
プロセッサ	検索	ふるい	パズル	Acker
iAPX 432 (8MHz)	4.4	978	45700	47800
80286 (8MHz)	1.4	168	9138	2218

- 検索：長さ 120 の文字列から長さ 15 の部分文字列を探す
- ふるい：素数を探索する（エラストテネスの篩）
- パズル：bin packing 問題（アイテムを詰め切れる最小の容器数を求める問題）
- Acker：Ackermann(3, 6) を計算
 - BuntanPC でも計測を試みたがプログラムが暴走してしまった。おそらく再帰が深すぎた。比較にちょうど良いと思ったんだけど。



- iAPX 432 とは
-  システム構成と Multibus
- 2 階層のアドレス変換
- 機械語の構造
- オブジェクト保護

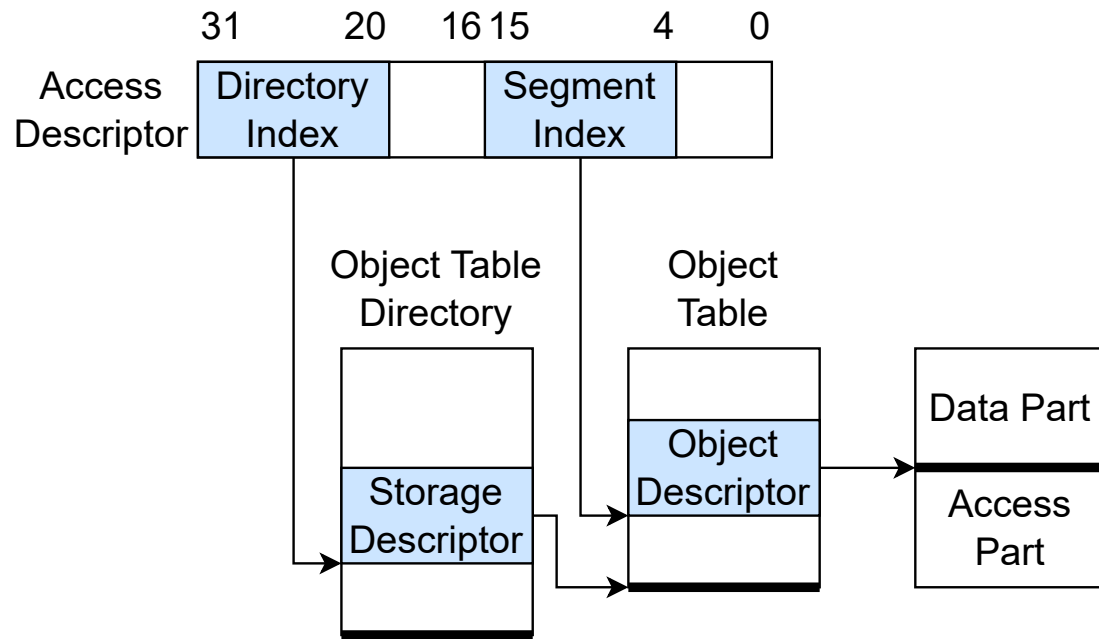




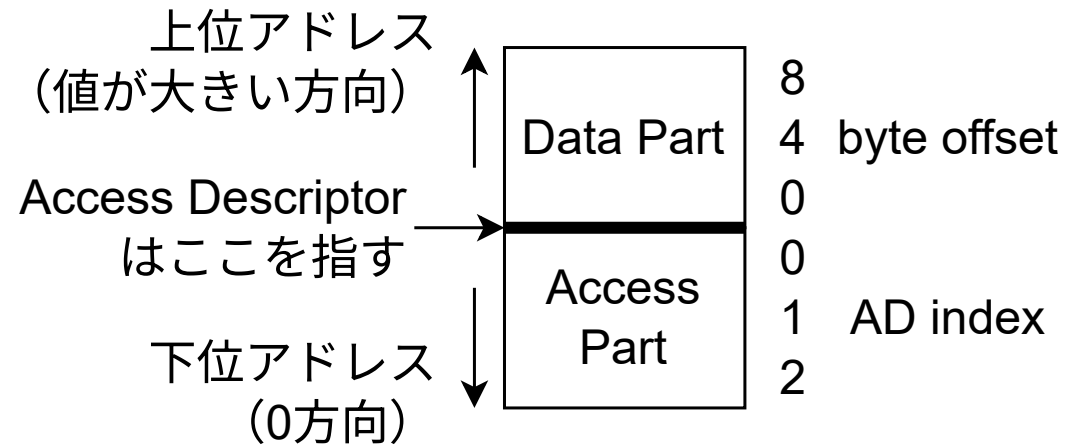


- iAPX 432 とは
- システム構成と Multibus
- ▶ 2 階層のアドレス変換
- 機械語の構造
- オブジェクト保護

2 階層のアドレス変換



- AD (Access Descriptor) は 2 つのインデックスを持つ
 - 2 階層ページングと類似
 - それぞれ 12 ビット = 4096 個を指せる
- Storage/Object descriptor は物理アドレスを持つ
- 1つのメモリ領域（オブジェクト）を指す Object Descriptor は高々1つ
- アドレス空間は $12+12+16=40$ ビット



Access Partが
ないオブジェクト



Data Partが
ないオブジェクト



- iAPX 432 のオブジェクトはすべて Data Part と Access Part を持つ
 - どちらかのサイズが 0 のこともある
- Data Part はデータ本体
- Access Part は AD が並ぶ
- それぞれの Part は最大 64KB

Domain Object

Public data variables	
Private data variables	
Fault instruction object AD	{ iAPX 432が規定
Trace instruction object AD	
Instruction object AD	{ コンパイラが規定
Instruction object AD	
Data constants AD	
Domain-local object AD	
...	

- 1つのモジュール≡プロセスを表現
 - モジュール：プロシージャ群+データ
- Fault/Trace instruction object AD は、例外発生時に実行する命令列を格納したオブジェクトを指す
- その他の AD はコンパイラが独自に生成
- 1つのプロシージャにつき 1つ以上の Instruction object に対応
 - 1つの Instruction object には 8KB までの命令を格納できる
 - 8KB 以上のプロシージャは複数の Instruction object で表現

Instruction Object

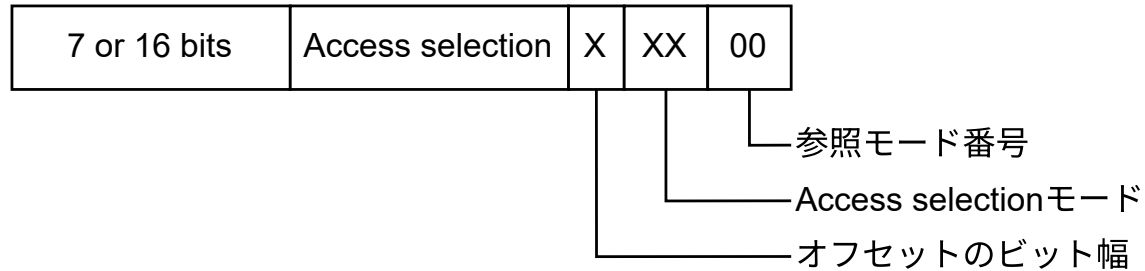
Instructions	byte offset
Local Constants DAI	8
Initial Operand Stack Pointer	6
Context Access Part Length	4
Context Data Part Length	2
	0

DAI: Domain Access Index
Domain objectのADを 指すインデックス

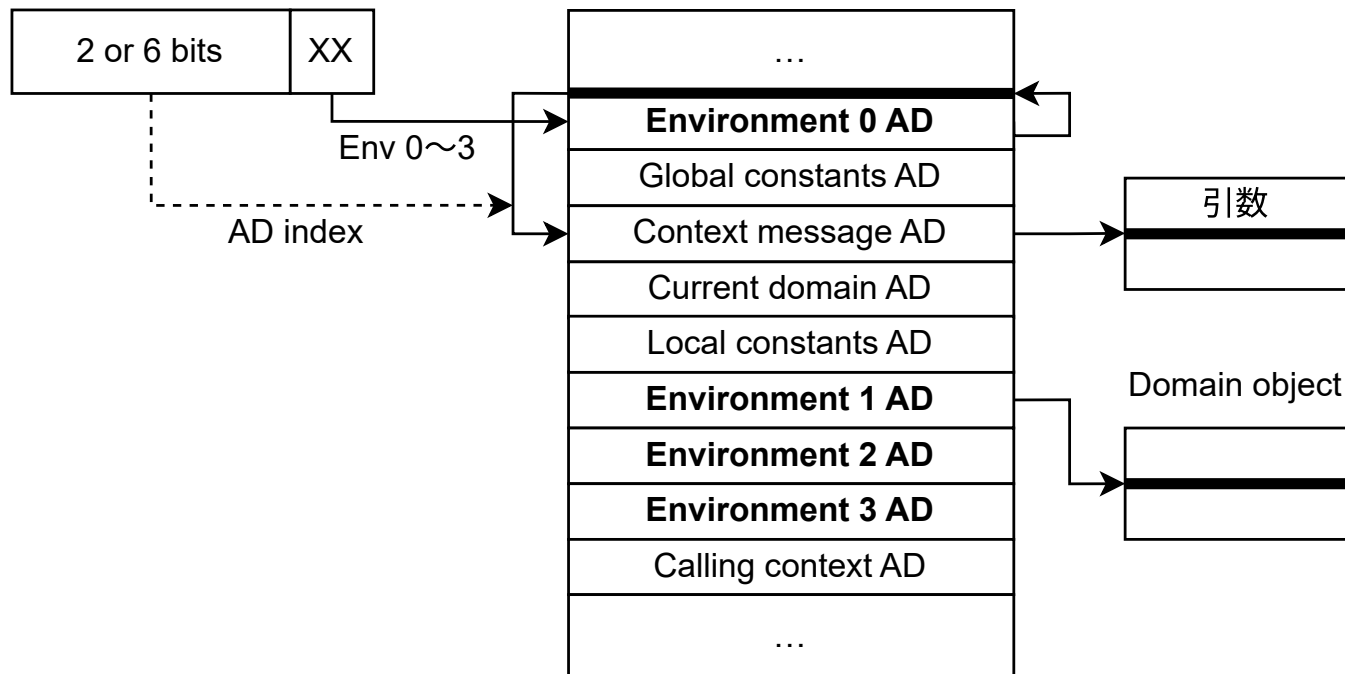
- 1つのプロシージャを表現
- Instructions は命令のビット列
 - このビット列はハフマン符号化（頻度に基づく圧縮）されている
- Local Constants DAI は定数が格納されたセグメントを指す
 - なんと 432 の命令は**即値を持ってない**！
 - メモリ領域に定数を置いておき、命令には「定数の場所を示す情報」を含める
- ビット単位でアドレッシングするので最大 64K bits = 8KB

スカラー値の参照だけでもメモリアクセスが多発

Scalar data reference




Access selectionモード
= 0/1のとき



- 2 ビットで Env 0~3 を選択
- 2/6 ビットでその Env 内の AD を選択
- スカラー値を得るだけで 4 回のメモリ読み込みが必要：
 - i. Env と AD index により指定される AD (32 ビット値) を読む
 - ii. AD が持つ 2 つのインデックスで 2 段階のアドレス変換を行う
 - iii. そのアドレスにオフセットを加えた場所を読む



- iAPX 432 とは
- システム構成と Multibus
- 2 階層のアドレス変換
-  機械語の構造
- オブジェクト保護

MSB				LSB	
次の命令	Opcode	Reference	Format	Class	前の命令

- それぞれ可変長のビット列
- Class がオペランド数とそれぞれのサイズを決定
- Format が参照とオペランドの対応を決定
- Reference が読み書きするデータの場所を決定


オペランド数	オペランドサイズ	Class
0	none	000110
1	byte	010110
1	double-byte	0000
...

よく使うクラスは
短いビット列

Format が参照とオペランドの対応を決定

オペランド数	オペランド1	オペランド2	オペランド3	明示的参照	Format
0				0	none
1	data ref 1			1	0
1	stack			0	1
2	data ref 1	data ref 2		2	00
2	data ref 1	data ref 1		1	10
2	data ref 1	stack		1	01
2	stack	data ref 1		1	011
2	stack	stack		0	111
3	data ref 1	data ref 2	data ref 3	3	0000
3



- iAPX 432 とは
- システム構成と Multibus
- 2 階層のアドレス変換
- 機械語の構造
-  オブジェクト保護

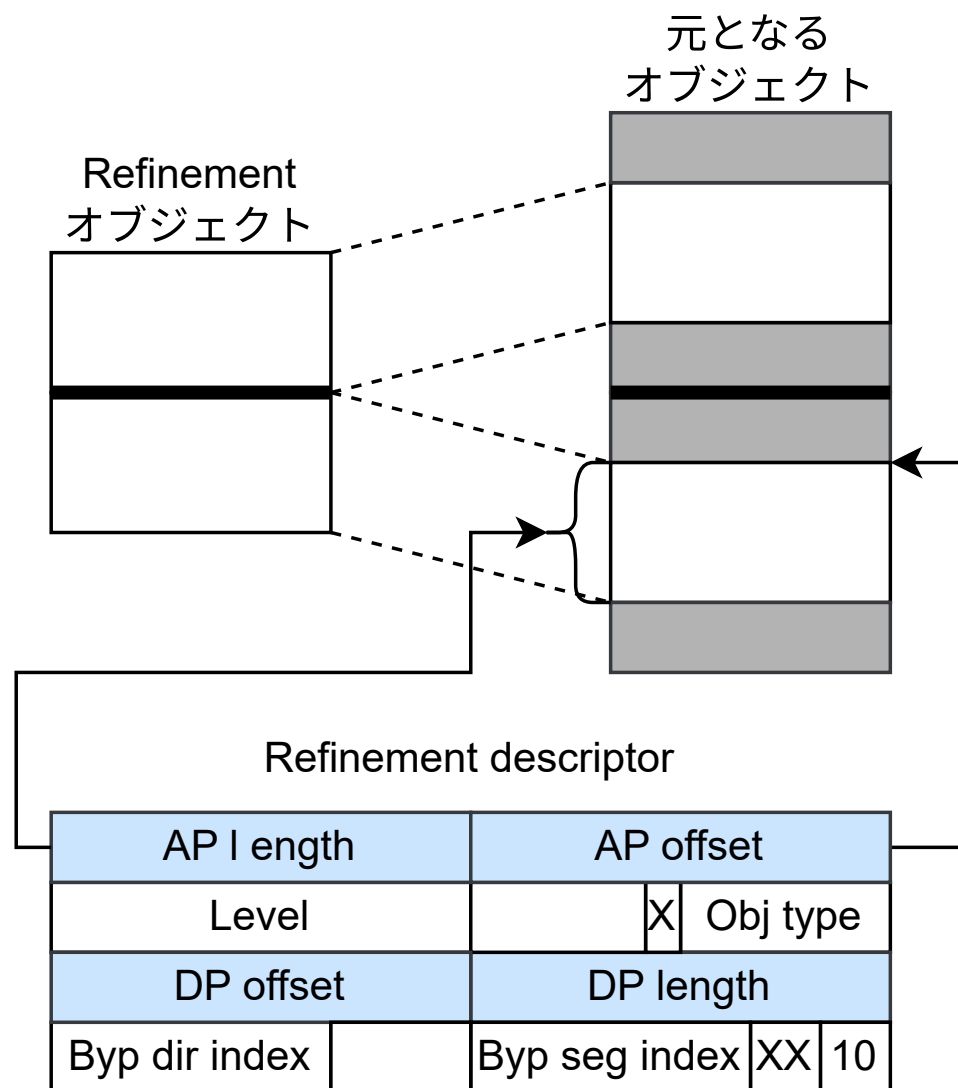
2 つの保護の仕組み

1. Refinement

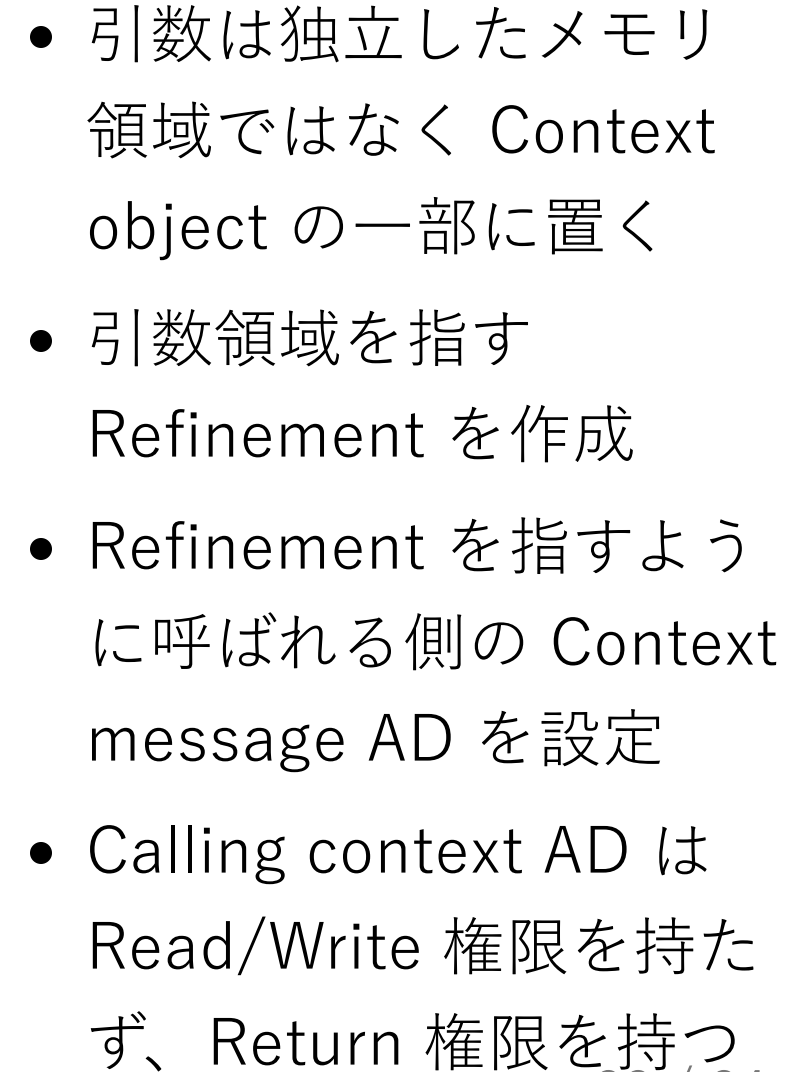
- あるオブジェクトの公開部分だけを見せるビュー
- ドメインに含まれる特定のプロシージャだけ公開する、というようなことが可能
- CREATE REFINEMENT という命令がある

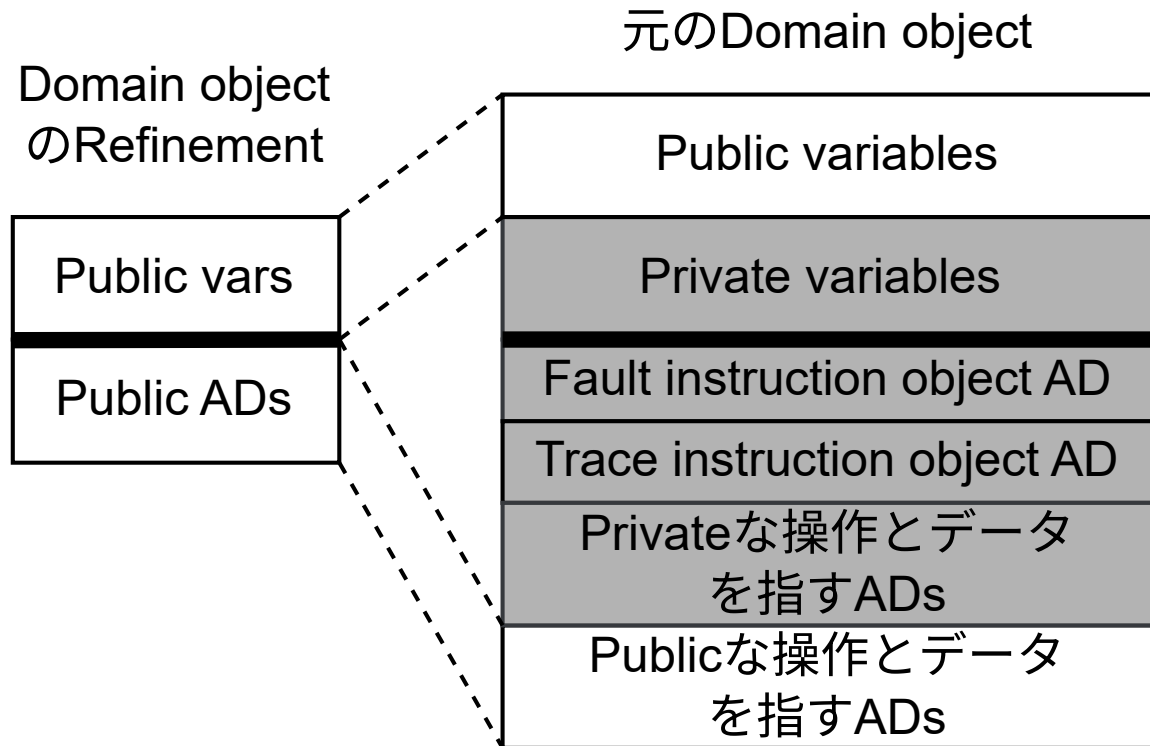
2. 制限と拡大 (restriction and amplification)

- 普段はオブジェクトへのアクセスを制限し、必要な場合に権限を拡大する
- RESTRICT RIGHTS / AMPLIFY RIGHTS という命令がある



- Refinement オブジェクトは元となるオブジェクトの一部を切り取ったビュー
 - 複数の非連続な場所は指定できない
- 使用例
 - プロシージャの引数だけにアクセス可能なRefinement を介して引数を渡す
 - Domain の一部のプロシージャのみを公開する





- Domain object は複数のプロシージャや変数を持つ
- Refinement により外部モジュールに対する可視性を制御できる
- 呼び出された公開プロシージャからは全てのプロシージャと変数が見える
 - Refinement を介して CALL が実行された場合でも、元の Domain object を指す AD が Context object に設定されるため



- iAPX 43201 iAPX 43202 VLSI General Data Processor (GDP のデータシート)
 - Intel (1981)
- iAPX432 General Data Processor Architecture Reference Manual
 - Intel (1981)
- Capability-Based Computer Systems の “Chapter 9: The Intel iAPX 432”
 - Henry M. Levy (1984)
- Intel iAPX 432 - Computer Science 460 - Final Project
 - David King, Liang Zhou, Jon Bryson, David Dickson (1999)
 - <http://www.brouhaha.com/~eric/retrocomputing/intel/iapx432/cs460/>