



簡易モデルの仕様:

- 1 word = 8 bit (= 1 byte) → 一度に扱うデータの大きさ
→ アキュムレータ (Acc)・命令レジスタ (IR)・演算回路・主記憶 1 番地分
 - 命令部: 3 bit、番地部: 5 bit
 - * 命令: $2^3 = 8$ つ以内 (load, store, add, subtract, jump, jump flag, stop)
 - 二進 3 桁の番号で扱う... 命令の符号化 (encoding)
- | 命令 | stop | load | store | add | subtract | jump | jump flag |
|-----|------|------|-------|-----|----------|------|-----------|
| 符号化 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
- * 番地: 0 番地から 31 番地まで
→ 主記憶 $2^5 = 32$ byte・プログラムカウンタ (PC) は 5 bit
 - フラグレジスタ (FR): 条件分岐の判断に用いる・ここでは Acc の符号 bit をコピー

処理の流れ: パルス発生器で、交互に 1 となる 2 つのパルス T_1, T_2 を発生

- $T_1 = 1$: 命令読込フェイズ
PC が指定する番地の主記憶の内容を読み出して IR に書込み・PC の値を 1 増やす
- $T_2 = 1$: 命令実行フェイズ
IR の内容の解析と実行 (命令に応じた所定のレジスタへの書込み)

命令の種類・実行:

- load: 主記憶から Acc への読出し
IR の番地部が指定する番地の主記憶の内容を Acc に書込・FR をセット (読み出した値が負なら FR を 1 に)
- store: Acc から主記憶への書込み
Acc の内容を IR の番地部が指定する番地の主記憶に書込み
- add / subtract: 演算 (加算 / 減算)
IR の番地部が指定する番地の主記憶の内容と Acc の内容とを演算回路で加算 / 減算し、Acc に書込・FR をセット (演算結果が負なら FR を 1 に)
- jump: 実行制御・無条件ジャンプ (次に実行する命令の番地を指定)
IR の番地部を PC に書込み
- jump flag: 実行制御・条件ジャンプ (FR が 1 のときのみジャンプ)
FR が 1 のときのみ、IR の番地部を PC に書込み
- stop: 実行の終了
IR の番地部を PC に書込み、パルスを止める