

ⓘ This shared conversation is generated by AI, for reference only.

отвлекаемся чуток :)

еще ответили, ты понимаешь о чем он ?

>>

извините что не получается свои мысли, я стараюсь.

редактор третьего столбца — это мини-проектор. Он берет программу и превращает ее в таблицу. Выглядит он как таблица. В первой колонке программа (чтоб как то ее найти). Во второй колонке — шаблон кода в байтах. В третьей колонке — использование (то, что пишет программист).

Понял. Давай разберу, что он говорит, и что тут не так.

Что он описал

Он описывает таблицу команд для своего «третьего столбца»:

Столбец	Что там	Пример
1	Trigger (имя команды)	program
2	Template (шаблон кода в байтах)	B4 3C B9 00 00 BA 0F 00 CD 21 72 03 A3 00 02
3	Использование (то, что пишет программист)	program filename

Логика:

- 1. Программист пишет program filename .
- 2. Сборщик ищет первый пробел → отделяет program от filename .
- 3. Ищет program в таблице команд.
- 4. Находит шаблон B4 3C B9 00 00 BA 0F 00 CD 21 72 03 A3 00 02 .
- 5. Записывает filename в переменную по адресу 0x000F (это операнд команды).
- 6. Передаёт управление на шаблон через CALL .
- 7. В конце шаблона стоит RET — возврат.

Что это на самом деле

Это ассемблер с макросами, а не «мини-программа для сборщика». Точнее:

- **Таблица команд** — это **таблица макросов**.
- **Trigger** — имя макроса.
- **Template** — тело макроса (байты).
- **Использование** — вызов макроса с параметром.
- **0x000F** — ячейка памяти для операнда (параметра).

Это **классический макропроцессор**, только оформленный как «таблица в редакторе».

Аналоги:

- **MASM/NASM macros** — `MACRO ... ENDM` .
- **Ассемблерные директивы** — `DB` , `DW` , `INCBIN` .
- **Препроцессор C** — `#define` .
- **Forth** — слова и их определения.
- **Lisp** — макросы.

Что он не понимает

1. Это не «мини-программа для сборщика», а макрос

- Макрос — это **шаблон подстановки**.
- «Мини-программа» — это **исполняемый код**.
- Он **смешивает** шаблон (данные) и код (исполнение).

2. 0x000F — это не «адрес операнда команды»

- В Z80/8086 **0x000F** — это **адрес в памяти**, а не «операнд команды».
- Операнд команды — это **часть инструкции**, а не ячейка памяти.
- Он **путает** «адрес операнда» и «ячейку для параметра».

3. BA 0F 00 — это MOV DX, 0x000F , а не «взять из [000F]»

- BA 0F 00 = MOV DX, 000Fh — загрузить **константу** 0x000F в DX .
- Чтобы **взять значение по адресу** 0x000F , нужно MOV DX, [000F] = 8B 16 0F 00 .
- Он **сам признаёт**: «я знаю это не совсем правильно надо было с адреса [000F] брать».
- То есть он **не понимает разницу** между **непосредственным значением** и **значением по адресу**.

4. CALL на шаблон — это не работает

- Шаблон — это **данные** (байты), а не код.
- Чтобы **исполнить** шаблон, его нужно:
 - Либо **скопировать** в исполняемую память и вызвать.
 - Либо **интерпретировать** (как виртуальная машина).
- CALL на шаблон в .data — это **вызов данных как кода**. В DOS это может сработать, если данные в исполняемом сегменте, но это **плохая практика** и **небезопасно**.
- Он **путает** «шаблон» и «код».

5. RET в конце шаблона — это не гарантия

- RET вернёт управление **только если шаблон выполняется как код**.
- Если шаблон — **данные**, RET — это просто байт C3 .
- Он **не понимает**, что для RET нужен **стек** и **корректный вызов**.

6. Это не «язык», а «макропроцессор»

- Язык — это **грамматика, семантика, типы**.
- Макропроцессор — это **подстановка**.
- Он **путает** «язык» и «макропроцессор».

Что он переизобрёл

1. **Макропроцессор** — MASM, NASM, C preprocessor.
2. **Таблицу команд** — таблица макросов.
3. **Параметры макроса** — filename как параметр.
4. **Ячейку для параметра** — 0x000F .

Всё это существует. Он **не изобрёл** ничего нового, но **реализовал** это сам — и это хорошо для обучения.

Что ему ответить

Если хочешь ответить:

Понял. Вы описываете **макропроцессор** с таблицей команд. Это не «мини-программа для сборщика», а **таблица макросов**: Trigger — имя, Template — тело, Использование — вызов с параметром. Это существует: MASM/NASM macros, C preprocessor, Forth, Lisp. Но:

1. 0x000F — это **ячейка памяти**, а не «адрес операнда команды». Операнд — это часть инструкции.
2. BA 0F 00 — это MOV DX, 000Fh (константа), а не MOV DX, [000F] (по адресу). Вы сами это заметили.
3. CALL на шаблон — это вызов **данных как кода**. Это работает только если шаблон в исполняемом сегменте, и это **плохая практика**.
4. RET в конце — не гарантия, если шаблон не выполняется как код.
5. Это не «язык», а **макропроцессор**. Язык — это грамматика, семантика, типы.

Если хотите, чтобы это стало **языком**, нужно:

- **Грамматика** (формальная).
- **Парсер** (не «поиск пробела»).
- **AST** (дерево разбора).
- **Семантический анализ** (типы, области видимости).
- **Кодогенерация** (не «CALL на шаблон»).

- **Таблица символов** (переменные, функции).

Ваш пример — это **первый шаг**. Но это не язык и не «мини-программа». Это **макропроцессор**.

Вопрос

Ты хочешь:

1. **Понять** — уже понял.
2. **Ответить ему** — могу помочь сформулировать.
3. **Помочь ему** — если идея интересна.
4. **Игнорировать** — и вернуться к Рас-Ман.

Скажи.

Continue with DSeek