УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Основы профессиональной деятельности»

Отчет

По лабораторной работе №5

Вариант 2

Студент

P3132

Преподаватель

Санкт-Петербург, 2025 г.

Текст задания:

Лабораторная работа №5

По выданному преподавателем варианту разработать программу асинхронного обмена данными с внешним устройством. При помощи программы осуществить ввод или вывод информации, используя в качестве подтверждения данных сигнал (кнопку) готовности ВУ.

Введите номер варианта 2

1. Программа осуществляет асинхронный вывод данных на ВУ-3
2. Программа начинается с адреса $4C4_{16}$. Размещаемая строка находится по адресу 608_{16}.
3. Строка должна быть представлена в кодировке КОИ-8.
4. Формат представления строки в памяти: АДР1: СИМВ2 СИМВ1 АДР2: СИМВ4 СИМВ3 ... СТОП_СИМВ.
5. Ввод или вывод строки должен быть завершен по символу с кодом 00 (NUL). Стоп символ является обычным символом строки и подчиняется тем же правилам расположения в памяти что и другие символы строки.

Код программы

0xF0C9 here this data represented as Symbol 2(F0) and 1(C9) (based on variant). So we need to send the symbols as an order. 1 2 3 4… First symbol is == C9.

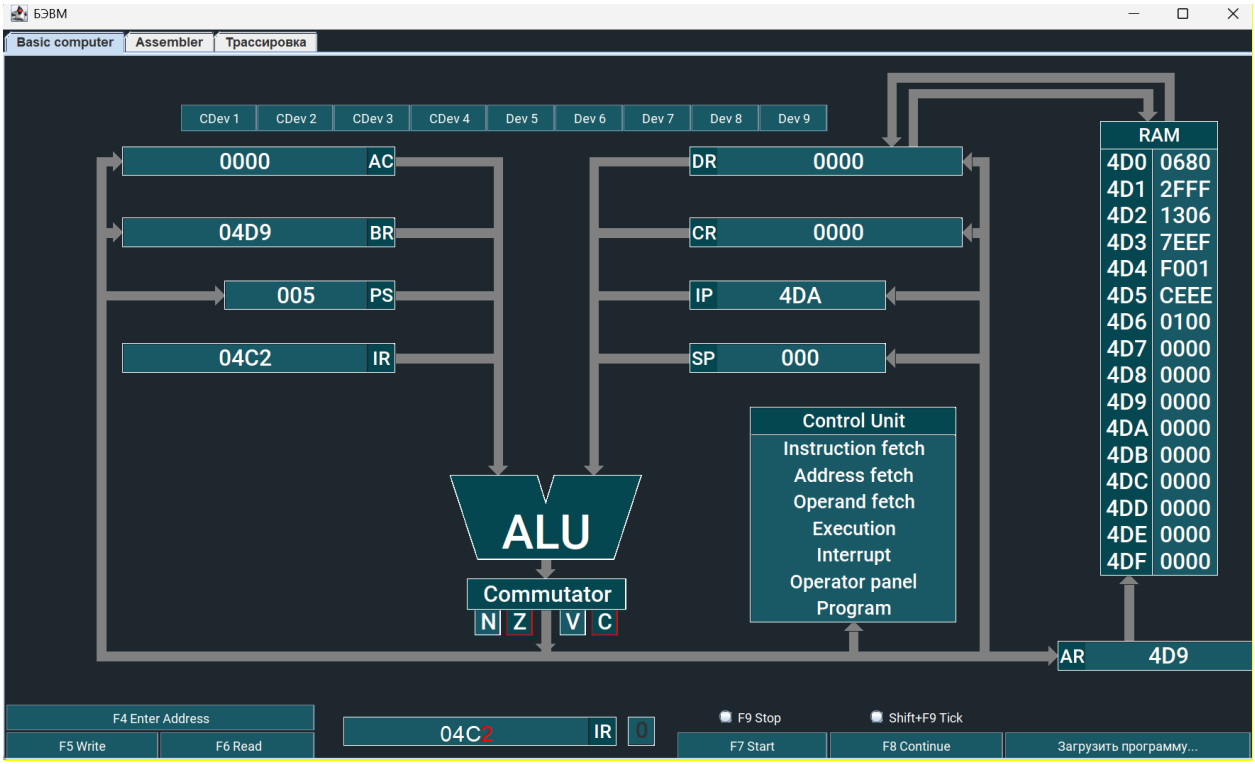
Адрес	Код команды	Мнемоника	Описание
4C2	0608	ADR	Address of the current element from the array
4C3	0000	STOP_SYMBL	Symbol at which the programme stops
4C4	1207	IN 7	Reading from the VU register (receiving a signal)
4C5	2F40	AND #40	Checking 6th bit for readiness
4C6	F0FD	BEQ 4C4	If it is not ready, go back to the beginning of the loop – 4C4 (loop till it become ready)
4C7	A8FA	LD M(4C2)	Loading an element (ADDRES: SYMBOL 2 1) 176 is a pointer here
4C8	2FFF	AND #FF	direct load FF with AND operation, ISOLATE FIRST 8 BITS
4C9	1306	OUT 6	Symbol 1 (first data) sent to VU(device)
4CA	7EF8	CMP (4C3)	Check for stop symbol, AC – 0000
4CB	F00A	BEQ 4D6	Branch (stop sending), if it's a stop sign
4CC	1207	IN 7	Reading from the VU register
4CD	2F40	AND #40	Checking 6th bit for readiness
4CE	F0FD	BEQ 4CC	If it is not ready, go back to the beginning
4CF	AAF2	LD(4C2)+	Loading an element ADDRES: SYMBOL 2 1 Loads the operand pointed 4C2 which is the element in address 608 than pointer +1
4D0	0680	SWAB	ADDRES: SYMBOL 1 2
4D1	2FFF	AND #FF	Isolate first 8 bits to send 2nd symbol
4D2	1306	OUT 6	Symbol 2 sent to VU
4D3	FEEF	CMP 175	Checking for stop symbol
4D4	F001	BEQ +1	Branch if it's a stop sign
4D5	CEEE	JUMP (4C4)	Go back to start of the program
4D6	0100	HLT	Finish the program
-----	-------	--------	----------
608	0xF0C9	ARRAY ELEMENTS	
609	0xD7DE		
60A	0x0A01		

Код программы на языке ассемблер

```
ORG 0x4C2
ADR: WORD $ARRAY
STOP_SYMBL: WORD 0x00

SYM1: IN 7
AND #0x40
BEQ SYM1
LD (ADR)
AND #0xFF
OUT 6
CMP STOP_SYMBL
BEQ STOP
SYM2: IN 7
AND #0x40
BEQ SYM2
LD (ADR)+
SWAB
AND #0xFF
OUT 6
CMP STOP_SYMBL
BEQ STOP
JUMP SYM1

STOP: HLT

ORG 0x608
ARRAY: WORD 0xF0C9, 0xD7DE, 0x0001
```

Word F0C9 would be = Пи … You can change the words. (coding table for your variant above is)

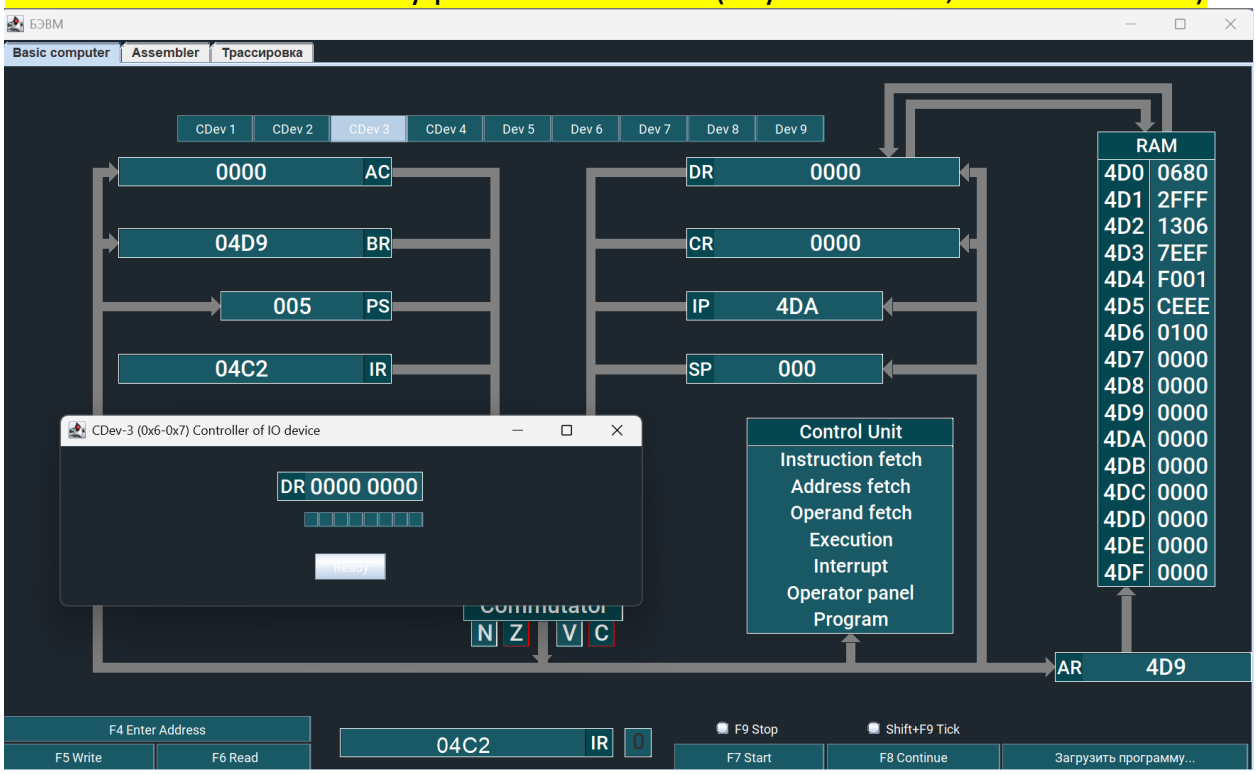
After you compile your assembler code, enter your address from the main screen(F4). Then press continue. Your program will start.



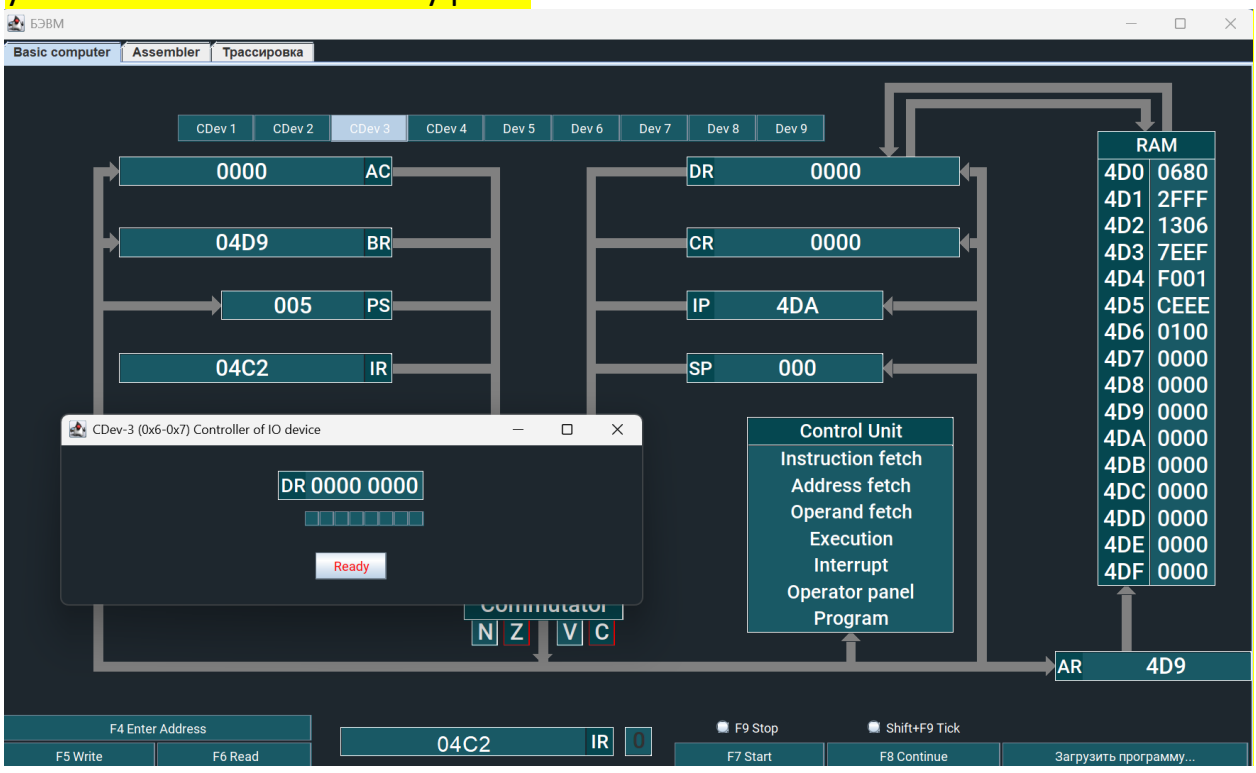
After that open the device 3 above. You will have small screen for device 3 here. In the command tablet here is a command for checking readiness of the device (

4C4	1207	IN 7	Reading from the VU register (receiving a signal)
4C5	2F40	AND #40	Checking 6th bit for readiness

). That readiness is the readiness button in the device 3. When you press it it will take data from the memory place of the bvem (in your variant, that is 608-60A)

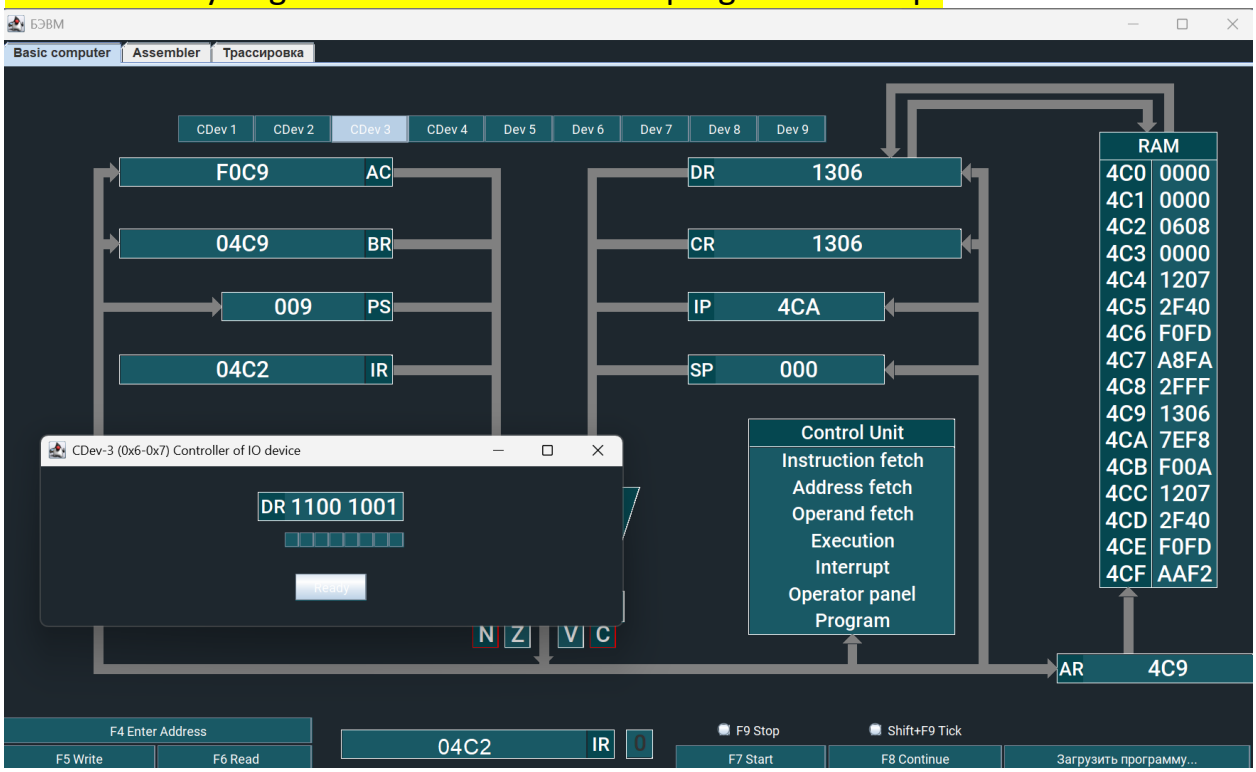


When you press ready it will turn red. Then you shoul continue to run your program till it get white again. After it become white, its not ready, it will have your value from the memory place



We got the first symbol == D9. Its in device right now. If you dont press ready and continue to run your program, It will stuck in loop till ytou press ready button. So

each time you got the value press ready and then continue. You will get the all values. After you get 00 from the last value program will stop.



Назначение программы (Program Purpose)

- Посимвольный асинхронный вывод строки из памяти БЭВМ на ВУ-3.
- Использование сигнала готовности ВУ-3 для синхронизации вывода каждого байта.
- Обработка строки в кодировке КОИ-8.
- Завершение работы при обнаружении символа конца строки (код 00).

Описание исходных данных (Input Data Description)

- ADR (адрес 4C2): Указатель на текущее обрабатываемое слово строки. Инициализируется адресом начала строки (608).
- STOP_SYMBL (адрес 4C3): Слово, содержащее код символа конца строки (0000).
- ARRAY (адрес 608): Массив 16-битных слов, содержащий выводимую строку символов.

Область представления (Data Representation) - ОПИ:

- Адреса (ADR, адреса в памяти): Беззнаковые 11-разрядные числа (000 – 7FF).
- Элемент строки (ARRAY): 16-битное слово, содержащее два 8-битных символа в кодировке КОИ-8.
    - Старший байт (биты 15-8): Первый символ слова (СИМВ1).
    - Младший байт (биты 7-0): Второй символ слова (СИМВ2).
- Символ конца строки (STOP_SYMBL): 16-битное слово 0000.

Область допустимых значений (Range of Acceptable Values) - ОДЗ:

- Адреса: Все адреса программы и данных должны быть в диапазоне 000 – 7FF.
- Строка: Должна содержать как минимум один байт 00 для корректного завершения.
- Состояние ВУ-3: Бит 6 регистра состояния ВУ-3 (порт 7) должен принимать значение 1 (готов) для продолжения вывода.
    - {0x4D6 ≤ ADR ≤ 0x7FF} - Указатель должен указывать на допустимую ячейку памяти.
    - (Содержимое_байта по ADR) ≠ 00 - Условие продолжения цикла вывода.
    - (Состояние флага ВУ-3, бит 6) == 1 - Условие для выполнения команды OUT.

Вывод

- В ходе выполнения лабораторной работы я изучил устройство ввода и вывода в БЭВМ и работу БЭВМ с устройствами по сигналам готовности. Также изучил построение кода на ассемблере.

Таблица трассировки

Адрес	Код	IP	CR	AR	DR	SP	BR	AC	NZVC	Адрес	Новый код
4C2	0608	4C3	0608	4C2	0608	000	04C2	0000	0100		
4C3	0000	4C4	0000	4C3	0000	000	04C3	0000	0100		
4C4	1207	4C5	1207	4C4	1207	000	04C4	0040	0100		
4C5	2F40	4C6	2F40	4C5	0040	000	0040	0040	0000		
4C6	F0FD	4C7	F0FD	4C6	F0FD	000	04C6	0040	0000		

4C7	A8FA	4C8	A8FA	608	F0C9	000	FFFA	F0C9	1000		
4C8	2FFF	4C9	2FFF	4C8	FFFF	000	FFFF	F0C9	1000		
4C9	1306	4CA	1306	4C9	1306	000	04C9	F0C9	1000		
4CA	7EF8	4CB	7EF8	4C3	0000	000	FFF8	F0C9	1001		
4CB	F00A	4CC	F00A	4CB	F00A	000	04CB	F0C9	1001		
4CC	1207	4CD	1207	4CC	1207	000	04CC	F040	1001		
4CD	2F40	4CE	2F40	4CD	0040	000	0040	0040	0001		
4CE	F0FD	4CF	F0FD	4CE	F0FD	000	04CE	0040	0001		
4CF	AAF2	4D0	AAF2	608	F0C9	000	FFF2	F0C9	1001	4C2	0609
4D0	0680	4D1	0680	4D0	0680	000	04D0	C9F0	1001		
4D1	2FFF	4D2	2FFF	4D1	FFFF	000	FFFF	C9F0	1001		
4D2	1306	4D3	1306	4D2	1306	000	04D2	C9F0	1001		
4D3	7EEF	4D4	7EEF	4C3	0000	000	FFEF	C9F0	1001		
4D4	F001	4D5	F001	4D4	F001	000	04D4	C9F0	1001		
4D5	CEEE	4C4	CEEE	4D5	04C4	000	FFEE	C9F0	1001		
4C4	1207	4C5	1207	4C4	1207	000	04C4	C940	1001		
4C5	2F40	4C6	2F40	4C5	0040	000	0040	0040	0001		
4C6	F0FD	4C7	F0FD	4C6	F0FD	000	04C6	0040	0001		
4C7	A8FA	4C8	A8FA	609	D7DE	000	FFFA	D7DE	1001		
4C8	2FFF	4C9	2FFF	4C8	FFFF	000	FFFF	D7DE	1001		
4C9	1306	4CA	1306	4C9	1306	000	04C9	D7DE	1001		
4CA	7EF8	4CB	7EF8	4C3	0000	000	FFF8	D7DE	1001		
4CB	F00A	4CC	F00A	4CB	F00A	000	04CB	D7DE	1001		
4CC	1207	4CD	1207	4CC	1207	000	04CC	D740	1001		
4CD	2F40	4CE	2F40	4CD	0040	000	0040	0040	0001		
4CE	F0FD	4CF	F0FD	4CE	F0FD	000	04CE	0040	0001		
4CF	AAF2	4D0	AAF2	609	D7DE	000	FFF2	D7DE	1001	4C2	060A
4D0	0680	4D1	0680	4D0	0680	000	04D0	DED7	1001		
4D1	2FFF	4D2	2FFF	4D1	FFFF	000	FFFF	DED7	1001		
4D2	1306	4D3	1306	4D2	1306	000	04D2	DED7	1001		
4D3	7EEF	4D4	7EEF	4C3	0000	000	FFEF	DED7	1001		
4D4	F001	4D5	F001	4D4	F001	000	04D4	DED7	1001		
4D5	CEEE	4C4	CEEE	4D5	04C4	000	FFEE	DED7	1001		
4C4	1207	4C5	1207	4C4	1207	000	04C4	DE40	1001		
4C5	2F40	4C6	2F40	4C5	0040	000	0040	0040	0001		
4C6	F0FD	4C7	F0FD	4C6	F0FD	000	04C6	0040	0001		
4C7	A8FA	4C8	A8FA	60A	0001	000	FFFA	0001	0001		
4C8	2FFF	4C9	2FFF	4C8	FFFF	000	FFFF	0001	0001		
4C9	1306	4CA	1306	4C9	1306	000	04C9	0001	0001		
4CA	7EF8	4CB	7EF8	4C3	0000	000	FFF8	0001	0001		
4CB	F00A	4CC	F00A	4CB	F00A	000	04CB	0001	0001		

4CC	1207	4CD	1207	4CC	1207	000	04CC	0040	0001		
4CD	2F40	4CE	2F40	4CD	0040	000	0040	0040	0001		
4CE	F0FD	4CF	F0FD	4CE	F0FD	000	04CE	0040	0001		
4CF	AAF2	4D0	AAF2	60A	0001	000	FFF2	0001	0001	4C2	060B
4D0	0680	4D1	0680	4D0	0680	000	04D0	0100	0001		
4D1	2FFF	4D2	2FFF	4D1	FFFF	000	FFFF	0100	0001		
4D2	1306	4D3	1306	4D2	1306	000	04D2	0100	0001		
4D3	7EEF	4D4	7EEF	4C3	0000	000	FFEF	0100	0001		
4D4	F001	4D5	F001	4D4	F001	000	04D4	0100	0001		
4D5	CEEE	4C4	CEEE	4D5	04C4	000	FFEE	0100	0001		
4C4	1207	4C5	1207	4C4	1207	000	04C4	0140	0001		
4C5	2F40	4C6	2F40	4C5	0040	000	0040	0040	0001		
4C6	F0FD	4C7	F0FD	4C6	F0FD	000	04C6	0040	0001		
4C7	A8FA	4C8	A8FA	60B	0000	000	FFFA	0000	0101		
4C8	2FFF	4C9	2FFF	4C8	FFFF	000	FFFF	0000	0101		
4C9	1306	4CA	1306	4C9	1306	000	04C9	0000	0101		
4CA	7EF8	4CB	7EF8	4C3	0000	000	FFF8	0000	0101		
4CB	F00A	4D6	F00A	4CB	F00A	000	000A	0000	0101		
4D6	0100	4D7	0100	4D6	0100	000	04D6	0000	0101		

Run your bvem in terminal to get the tracing table.
Run this command : java -jar -Dmode=dual bcomp-ng.jar
That command automatically open the bvem, and in that bvem you should run your brogram shift+f9 format. Then each step will appear in terminal (cmd). I already did it. I wrote the way how to get tracing, for you to understand.