VIRTUAL16 Instruction Set

	1 st Byte		2 nd Byte		3 rd Byte		
	1st Nb (BIT7:4)	2 nd Nb (BIT3:0)	1st Nb (BIT7:4)	2 nd Nb (BIT3:0)	1st Nb (BIT7:4)	2 nd Nb (BIT3:0)	See
RET	0	0	0	0	-	-	1
MOV Rs, Rd	0	1	Rs	Rd	-	-	
MOV @Rs, Rd	0	2	Rs	Rd	-	-	8
MOV Rs, @Rd	0	3	Rs	Rd	-	-	8
MOV +@Rs, Rd	0	4	Rs	Rd	-	-	8
MOV Rs, +@Rd	0	5	Rs	Rd	-	-	8
PUSH Rd	0	6	Rd	X	-	-	6
POP Rd	0	7	Rd	X	-	-	6
CLR Rd	0	8	Rd	X	-	-	6
SWAP Rs, Rd	0	9	Rs	Rd	-	-	2
AND Rs, Rd	0	Α	Rs	Rd	-	-	
OR Rs, Rd	0	В	Rs	Rd	-	-	
XOR Rs, Rd	0	С	Rs	Rd	-	-	
JSR ADDR	0	D	ADDR[7:4]	ADDR[3:0]	ADDR[15:12]	ADDR[11:8]	3, 7
RTS	0	E	-	-	-	-	7
JMP ADDR	0	F	ADDR[7:4]	ADDR[3:0]	ADDR[15:12]	ADDR[11:8]	3, 7
	1 st Byte		2 nd Byte		3 rd Byte		
	1 St [Dvto.	n d l	Dyto	ord I	Dyto	
		•		•			500
INC Dd. #COLINT	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	1st Nb (BIT7:4)	2 nd Nb (BIT3:0)	3 rd I 1 st Nb (BIT7:4)	Byte 2 nd Nb (BIT3:0)	See
INC Rd, #COUNT	1 st Nb (BIT7:4) 1	2 nd Nb (BIT3:0) 0	1 st Nb (BIT7:4) Rd	2 nd Nb (BIT3:0) COUNT			4
DEC Rd, #COUNT	1 st Nb (BIT7:4) 1 1	2 nd Nb (BIT3:0) 0 1	1 st Nb (BIT7:4) Rd Rd	2 nd Nb (BIT3:0) COUNT COUNT			
DEC Rd, #COUNT ADD Rs, Rd	1 st Nb (BIT7:4) 1 1 1	2 nd Nb (BIT3:0) 0 1 2	1 st Nb (BIT7:4) Rd Rd Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd			4
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd	1 st Nb (BIT7:4) 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3	1 st Nb (BIT7:4) Rd Rd Rs Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd	1 st Nb (BIT7:4) - - - -		4
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd	1 st Nb (BIT7:4) 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4	1 st Nb (BIT7:4) Rd Rd Rs Rs Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd	1 st Nb (BIT7:4) - - - -		4
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd	1 st Nb (BIT7:4) 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5	1 st Nb (BIT7:4) Rd Rd Rs Rs Rs Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	4
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6	1 st Nb (BIT7:4) Rd Rd Rs Rs Rs Rs Rs Rs Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd Rd Rd Rd	1 st Nb (BIT7:4)		10
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd UMUL Rs, Rd	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6 7	1 st Nb (BIT7:4) Rd Rd Rs Rs Rs Rs Rs Rs Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd Rd Rd	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	10 10
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd UMUL Rs, Rd ASL Rd, #COUNT	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6 7	1 st Nb (BIT7:4) Rd Rd Rs Rs Rs Rs Rs Rs Rs Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd Rd Rd Rd	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	10 10 4
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd UMUL Rs, Rd ASL Rd, #COUNT ASR Rd, #COUNT	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6 7 8 9	1 st Nb (BIT7:4) Rd Rd Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd Rd COUNT	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	10 10 4 4
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd UMUL Rs, Rd ASL Rd, #COUNT ASR Rd, #COUNT LSL Rd, #COUNT	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6 7 8 9 A	1st Nb (BIT7:4) Rd Rd Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd Rd COUNT COUNT	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	10 10 4 4
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd UMUL Rs, Rd ASL Rd, #COUNT ASR Rd, #COUNT LSL Rd, #COUNT LSR Rd, #COUNT	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6 7 8 9 A B	1st Nb (BIT7:4) Rd Rd Rs	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd Rd COUNT COUNT COUNT COUNT COUNT	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	10 10 4 4 4
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd UMUL Rs, Rd UMUL Rs, Rd LSE Rd, #COUNT ASR Rd, #COUNT LSE Rd, #COUNT LSE Rd, #COUNT ROL Rd, #COUNT	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6 7 8 9 A B C	1st Nb (BIT7:4) Rd Rd Rs Rs Rs Rs Rs Rs Rs Rs Rs Rd Rd Rd Rd	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd COUNT COUNT COUNT COUNT COUNT COUNT COUNT	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	10 10 4 4 4 4 4,9
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd UMUL Rs, Rd ASL Rd, #COUNT ASR Rd, #COUNT LSL Rd, #COUNT LSR Rd, #COUNT ROL Rd, #COUNT ROL Rd, #COUNT	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6 7 8 9 A B C D	1st Nb (BIT7:4) Rd Rd Rs Rs Rs Rs Rs Rs Rs Rs Rd Rd Rd Rd Rd	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd COUNT	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	10 10 4 4 4 4 4,9 4,9
DEC Rd, #COUNT ADD Rs, Rd ADC Rs, Rd SUB Rs, Rd SBC Rs, Rd SMUL Rs, Rd UMUL Rs, Rd UMUL Rs, Rd ASL Rd, #COUNT ASR Rd, #COUNT LSL Rd, #COUNT ROL Rd, #COUNT	1 st Nb (BIT7:4) 1 1 1 1 1 1 1 1 1 1 1 1 1	2 nd Nb (BIT3:0) 0 1 2 3 4 5 6 7 8 9 A B C	1st Nb (BIT7:4) Rd Rd Rs Rs Rs Rs Rs Rs Rs Rs Rs Rd Rd Rd Rd	2 nd Nb (BIT3:0) COUNT COUNT Rd Rd Rd Rd Rd COUNT COUNT COUNT COUNT COUNT COUNT COUNT	1 st Nb (BIT7:4)	2 nd Nb (BIT3:0)	10 10 4 4 4 4 4,9

VIRTUAL16 Instruction Set (continued)

	1 st Byte		2 nd Byte		3 rd Byte		
	1st Nb (BIT7:4)	2 nd Nb (BIT3:0)	1st Nb (BIT7:4)	2 nd Nb (BIT3:0)	1st Nb (BIT7:4)	2 nd Nb (BIT3:0)	See
MOVL Rd, #VAL	2	Rd	VAL[7:4]	VAL[3:0]	-	-	
MOVH Rd, #VAL	3	Rd	VAL[7:4]	VAL[3:0]	-	-	
MOV Rd, #VAL	4	Rd	VAL[7:4]	VAL[3:0]	VAL[15:12]	VAL[11:8]	7
BPL Rs, OFFSET	5	Rs	OFFSET[7:4]	OFFSET[3:0]	-	-	5
BMI Rs, OFFSET	6	Rs	OFFSET[7:4]	OFFSET[3:0]	-	-	5
BEQ Rs, OFFSET	7	Rs	OFFSET[7:4]	OFFSET[3:0]	-	-	5
BNE Rs, OFFSET	8	Rs	OFFSET[7:4]	OFFSET[3:0]	-	-	5
BM1 Rs, OFFSET	9	Rs	OFFSET[7:4]	OFFSET[3:0]	-	-	5
BNM1 Rs, OFFSET	Α	Rs	OFFSET[7:4]	OFFSET[3:0]	-	-	5
BRA OFFSET	В	X	OFFSET[7:4]	OFFSET[3:0]	-	-	5, 6
BCS OFFSET	С	X	OFFSET[7:4]	OFFSET[3:0]	-	-	5, 6
BCC OFFSET	D	X	OFFSET[7:4]	OFFSET[3:0]	-	-	5, 6
MOV ADDR, Rd	E	Rd	ADDR[7:4]	ADDR[3:0]	ADDR[15:12]	ADDR[11:8]	7
MOV Rs, ADDR	F	Rs	ADDR[7:4]	ADDR[3:0]	ADDR[15:12]	ADDR[11:8]	7

- 1: RET instruction must be last instruction always
- 2: SWAP instruction have 2 modes. If source and destination register is different it swaps register contents. If source and destination register is same it swaps LSB and MSB of register
- 3: JSR and JMP calculates destination address automatically. Assemble your VIRTUAL16 code starting from 0x0000 address for proper calculation
- 4: #COUNT must be "real count 1". If #COUNT == 0 VIRTUAL16 calculates it as 1, if #COUNT == 15 VIRTUAL16 calculates it as 16 etc.
- 5: OFFSET formula is = New Address Current Address 2. Same as 6502. Except BRA, BCS and BCC instructions you must give Register to test
- 6: x means don't matter. You can give any value to this nibbles
- 7: Except these instructions all other instructions are 16-bit wide
- 8: @ means register holds pointer. +@ means register holds pointer and after execution pointer will incremented by 2. These instructions destroy BITO of pointer. It means you cannot access odd locations in memory
- 9: Difference between ROL and RLC is ROL ignores carry for first rotate but RLC don't. Same for ROR and RRC.
- 10: SMUL and UMUL instructions are 16x16=32-bit and result held by R12 (LSW) and R13 (MSW)