VIRTUAL16 Instruction Set

	1st Byte		2nd Byte		3rd Byte		
	1st Nb (BIT7:4)	2nd Nb (BIT3:0)	1st Nb (BIT7:4)	2nd Nb (BIT3:0)	1st Nb (BIT7:4)	2nd 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	A	Rs	Rd	-	-	
OR Rs, Rd	0	B	Rs	Rd	-	-	
XOR Rs, Rd	0	C	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

	1st Byte		2nd Byte		3rd Byte		
	1st Nb (BIT7:4)	2nd Nb (BIT3:0)	1st Nb (BIT7:4)	2nd Nb (BIT3:0)	1st Nb (BIT7:4)	2nd Nb (BIT3:0)	See
INC Rd, #COUNT	1	0	Rd	COUNT	-	-	4
DEC Rd, #COUNT	1	1	Rd	COUNT	-	-	4
ADD Rs, Rd	1	2	Rs	Rd	-	-	
ADC Rs, Rd	1	3	Rs	Rd	-	-	
SUB Rs, Rd	1	4	Rs	Rd	-	-	
SBC Rs, Rd	1	5	Rs	Rd	-	-	
SMUL Rs, Rd	1	6	Rs	Rd	-	-	10
UMUL Rs, Rd	1	7	Rs	Rd	-	-	10
ASL Rd, #COUNT	1	8	Rd	COUNT	-	-	4
ASR Rd, #COUNT	1	9	Rd	COUNT	-	-	4
LSL Rd, #COUNT	1	A	Rd	COUNT	-	-	4
LSR Rd, #COUNT	1	B	Rd	COUNT	-	-	4
ROL Rd, #COUNT	1	C	Rd	COUNT	-	-	4, 9
RLC Rd, #COUNT	1	D	Rd	COUNT	-	-	4, 9
ROR Rd, #COUNT	1	E	Rd	COUNT	-	-	4, 9
RRC Rd, #COUNT	1	F	Rd	COUNT	-	-	4, 9

VIRTUAL16 Instruction Set (continued)

	1st Byte		2nd Byte		3rd Byte		
	1st Nb (BIT7:4)	2nd Nb (BIT3:0)	1st Nb (BIT7:4)	2nd Nb (BIT3:0)	1st Nb (BIT7:4)	2nd 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	A	Rs	OFFSET[7:4]	OFFSET[3:0]	-	-	5
BRA OFFSET	B	x	OFFSET[7:4]	OFFSET[3:0]	-	-	5, 6
BCS OFFSET	C	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 BIT0 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)