CODE	PARAMETER	DO
x00	xXX	MOV regs[0] -> memory[XX]
x01	xXX	MOV regs[1] -> memory[XX]
x02	xXX	MOV regs[2] -> memory[XX]
x03	xXX	MOV regs[3] -> memory[XX]
x04	xXX	MOV memory[XX] -> regs[0]
x05	xXX	MOV memory[XX] -> regs[1]
x06	xXX	MOV memory[XX] -> regs[2]
x07	xXX	MOV memory[XX] -> regs[3]
x08	xXX	MOV XX-> regs[0]
x09	xXX	MOV XX-> regs[1]
x0A	xXX	MOV XX-> regs[2]
x0B	xXX	MOV XX-> regs[3]
x0C	xXY	MOV regs[X] -> regs[Y]
x0D	xXY	MOV memory[regs[X]] -> regs[Y]
x0E	xXY	MOV regs[X] -> memory[regs[Y]]
x10	xXX	regs[0] = ~regs[0]
x11	xXX	regs[1] = ~regs[1]
x12	xXX	regs[2] = ~regs[2]
x13	xXX	regs[3] = ~regs[3]
x20	xXX	regs[0] &= XX
x21	xXX	regs[1] &= XX
x22	xXX	regs[2] &= XX
x23	xXX	regs[3] &= XX
x24	xXY	regs[Y] &= regs[X]
x30	xXX	regs[0] |= XX
x31	xXX	regs[1] |= XX
x32	xXX	regs[2] |= XX
x33	xXX	regs[3] |= XX
x34	xXY	regs[Y] |= regs[X]

x40	xXX	regs[0] ^= XX
x41	xXX	regs[1] ^= XX
x42	xXX	regs[2] ^= XX
x43	xXX	regs[3] ^= XX
x44	xXY	regs[Y] ^= regs[X]
x50	xXX	regs[0] = regs[0] << XX
x51	xXX	regs[1] = regs[1] << XX
x52	xXX	regs[2] = regs[2] << XX
x53	xXX	regs[3] = regs[3] << XX
x54	xXY	regs[Y] = regs[Y] << regs[X]
x60	xXX	regs[0] = regs[0] >> XX
x61	xXX	regs[1] = regs[1] >> XX
x62	xXX	regs[2] = regs[2] >> XX
x63	xXX	regs[3] = regs[3] >> XX
x64	xXY	regs[Y] = regs[Y] >> regs[X]
x70	xXX	regs[0] += XX
x71	xXX	regs[1] += XX
x72	xXX	regs[2] += XX
x73	xXX	regs[3] += XX
x74	xXY	regs[Y] += regs[X]
x80	xXX	regs[0] -= XX
x81	xXX	regs[1] -= XX
x82	xXX	regs[2] -= XX
x83	xXX	regs[3] -= XX
x84	xXY	regs[Y] -= regs[X]
x90	xXX	regs[0] *= XX
x91	xXX	regs[1] *= XX
x92	xXX	regs[2] *= XX
x93	xXX	regs[3] *= XX
x94	xXY	regs[Y] *= regs[X]

xA0	xXX	regs[0] /= XX
xA1	xXX	regs[1] /= XX
xA2	xXX	regs[2] /= XX
xA3	xXX	regs[3] /= XX
xA4	xXY	regs[Y] /= regs[X]
xB0	xXX	regs[0] %= XX
xB1	xXX	regs[1] %= XX
xB2	xXX	regs[2] %= XX
xB3	xXX	regs[3] %= XX
xB4	xXY	regs[Y] %= regs[X]
xC0	xXX	JUMP to line XX
xC1	xXX	JZ to line XX
xC2	xXX	JNZ to line XX
xC3	xXX	JS to line XX
xC4	xXX	JNS to line XX
xC5	xXX	JUMP if read (JR) to line XX
xC6	xXX	JUMP if not read (JNR) to line XX
xD0	xXX	digitalWrite(XX,LOW)
xD1	xXX	digitalWrite(XX,HIGH)
xD2	xXX	digitalRead(XX) (to use with JR/JNR)
xE0	xXX	pinMode(XX,INPUT)
xE1	xXX	pinMode(XX,OUTPUT)
	x000	program's end
	x001	regs[1] = analogueRead(regs[0])
	x002	analogueWrite(regs[0],regs[1])
	x003	delay(regs[0])
	x004	regs[1] = eeprom_read_byte(regs[0])
	x005	eeprom_write_byte(regs[0],regs[1])
	x006	Serial.print(string in the code line regs[0])

xF	x007	lcd.print(string in the code line regs[0])
	x008	Serial.begin(regs[0])
	x009	Serial.end()
	x00A	Serial.print(char in regs[0])
	x00B	Serial.print(int in regs[0])
	x00C	Serial.print(long in regs[0] << 16 | regs[1])
	x00D	Serial.print(unsigned int in regs[0])
	x00E	Serial.print(unsigned long in regs[0] << 16 | regs[1])
	x00F	micro() -> long stocked in regs[0] & regs[1]
	x010	millis() -> long stocked in regs[0] & regs[1]
	x011	long(regs[0] << 16 | regs[1]) - long(regs[2] << 16 | regs[3]) -> long stocked in regs[0] & regs[1]
	x012	lcd.begin()
	x013	lcd.end()
	x014	lcd.setCursor(regs[0] >> 4,regs[0] & 0x000F)
	x015	lcd.print(char in regs[0])
	x016	lcd.print(int in regs[0])
	x017	lcd.print(long in regs[0] << 16 | regs[1])
	x018	lcd.print(unsigned int in regs[0])
	x019	lcd.print(unsigned long in regs[0] << 16 | regs[1])
	x01A	lcd.clear()
	x01B	regs[1] = eeprom_read_word(regs[0])
	x01C	eeprom_write_word(regs[0],regs[1])
	x01D	lcd.print(unsigned int in regs[0],HEX)