0x00 NOP no operation
0x10 LDAi load accumulator A, immediate value
0x11 LDAa load accumulator A, absolute ref'd address
0x12 LDAx load accumulator A, indexed
0x14 STAa store accumulator A, absolute ref'd address
0x15 STAx store accumulator A, indexed
0x18 LDBi load accumulator B, immediate value
0x19 LDBa load accumulator B, absolute ref'd address
0x1A LDBx load accumulator B, indexed
0x1C STBa store accumulator B, absolute ref'd address
0x1D STBx store accumulator B, indexed
0x20 AND bitwise AND of accumulators A & B, result in A
0x21 OR bitwise OR of accumulators A & B, result in A
0x22 XOR bitwise XOR of accumulators A & B, result in A
0x23 COMA accumulator A bitwise complement
0x24 COMB accumulator A bitwise complement
0x25 ROLA rotate accumulator A bits left
0x26 RORA rotate accumulator A bits right
0x27 ROLB rotate accumulator B bits left
0x28 RORB rotate accumulator B bits right
0x29 SWAP swap values between accumulators A & B
0x2A LSL logical shift left through both accumulators
0x2B LSR logical shift right through both accumulators
0x30 CLRS clear status
0x31 SETS set status
0x32 SETC set carry
0x33 CLC clear carry
0x34 CLV clear overflow
0x35 BITAi memory contents AND acc A, immediate, only status affected
0x36 BITAa memory contents AND acc A, absolute, only status affected
0x37 BITAx memory contents AND acc A, indexed, only status affected
0x39 BITBi memory contents AND acc B, immediate, only status affected
0x3A BITBa memory contents AND acc B, absolute, only status affected
0x3B BITBx memory contents AND acc B, indexed, only status affected
0x40 PUSHXA push accumulator A onto Auxiliary Stack
0x41 POPXA pop accumulator A from Auxiliary Stack
0x42 PUSHXB push accumulator B onto Auxiliary Stack
0x43 POPXB pop accumulator B from Auxiliary Stack
0x44 SWAPS swap top 2 values of Auxiliary Stack a b c => b a c
0x45 DUP duplicate top value of Auxiliary Stack a b c => a a b c
0x46 OVER push copy of 2nd value of Auxiliary Stack on top // a b c => b a b c
0x47 ROT rotate top 3 values of Auxiliary Stack a b c => c a b
0x48 DROP delete top valueof Auxiliary Stack a b c => b c
0x49 TUCK tuck copy of top value of Auxiliary Stack below 2nd value a b c => a b a c
0x50 LSPi load Stack Pointer, immediate
0x51 LSPa load Stack Pointer, absolute
0x52 LSPx load Stack Pointer, indexed
0x54 LDXi load Index Register, immediate
0x55 LDXa load Index Register, absolute
0x56 LDXx load Index Register, indexed
0x58 SPCa store stack pointer, absolute
0x59 SPCx store stack pointer, indexed
0x5A PUSHA push accumulator A onto PC stack
0x5B POPA pop accumulator A onto PC stack
0x5C PUSHB push accumulator B onto PC stack
0x5D POPB pop accumulator B onto PC stack
0x60 JMPi immediate jump
0x61 JMPa absolute jump
0x63 BRA relative jump
0x64 JSRa jump to subroutine absolute
0x65 JSRr jump to subroutine relative (BSR)
0x66 RTS return from subroutine
0x68 BZS branch if zero set (6800 BEQ)
0x69 BZC branch if zero clear(BNE)
0x6A BCS branch if carry set
0x6B BCC branch if carry clear
0x6C BNS branch if negative set (BMI)
0x6D BNS branch if negative clear (BMI)
0x6E BVS branch if overflow set
0x6F BVC branch if overflow clear
0x70 BGE branch if greater than or equal to 0
0x71 BGT branch if greater than 0
0x72 BLT branch if less than 0
0x80 ABA add B to A
0x81 ADDAi add to accumulator A, immediate
0x82 ADDAa add to accumulator A, absolute
0x83 ADDAx add to accumulator A, indexed
0x85 ADDBi add to accumulator B, immediate
0x86 ADDBa add to accumulator B, absolute
0x87 ADDBx add to accumulator B, indexed
0x88 ADDBx to accumulator B, doubly-indexed
0x89 ADCAi add with carry, accumulator A, immediate
0x8A ADCAa add with carry, accumulator A, absolute
0x8B ADCAx add with carry, accumulator A, indexed

0x8D ADCBi add with carry, accumulator B, immediate
0x8E ADCBa add with carry, accumulator B, absolute
0x8F ADCBx add with carry, accumulator B, indexed
0x90 ADCBx add with carry, accumulator B, doubly-indexed
0x91 SUBAi add to accumulator A, immediate
0x92 SUBAa add to accumulator A, absolute
0x93 SUBAx add to accumulator A, indexed
0x95 SUBBi add to accumulator B, immediate
0x96 SUBBa add to accumulator B, absolute
0x97 SUBBx add to accumulator B, indexed
0x98 SUBBx to accumulator B, doubly-indexed
0xA0 ANDAi AND immediate memory contents with accumulator A, result in A
0xA1 ANDAa AND absolute memory contents with accumulator A, result in A
0xA2 ANDAx AND indexed memory contents with accumulator A, result in A
0xA4 ANDBi AND immediate memory contents with accumulator B, result in B
0xA5 ANDBa AND absolute memory contents with accumulator B, result in B
0xA6 ANDBx AND indexed memory contents with accumulator B, result in B
0xA8 ORAi OR immediate memory contents with accumulator A, result in A
0xA9 ORAa OR absolute memory contents with accumulator A, result in A
0xAA ORAx OR indexed memory contents with accumulator A, result in A
0xAC ORBi OR immediate memory contents with accumulator B, result in B
0xAD ORBa OR absolute memory contents with accumulator B, result in B
0xAE ORBx OR indexed memory contents with accumulator B, result in B
0xB0 EORAi EXOR immediate memory contents with accumulator A, result in A
0xB1 EORAa EXOR absolute memory contents with accumulator A, result in A
0xB2 EORAx EXOR absolute memory contents with accumulator A, result in
0xB4 EORBi EXOR immediate memory contents with accumulator B, result in B
0xB5 EORBa EXOR absolute memory contents with accumulator B, result in B
0xB6 EORBx EXOR indexed memory contents with accumulator B, result in B
0xB8 CAB compare A and B, only status flags affected
0xB9 CMPAi compare immediate memory and accumulator A, only status flags affected
0xBA CMPAa compare absolute memory and accumulator A, only status flags affected
0xBB CMPAx compare indexed memory and accumulator A, only status flags affected
0xBD CMPBi compare immediate memory and accumulator B, only status flags affected
0xBE CMPBa compare absolute memory and accumulator B, only status flags affected
0xBF CMPBx compare indexed memory and accumulator B, only status flags affected
0xC1 CLRA clear value of accumulator A
0xC2 CLRB clear value of accumulator B
0xC3 CLRa clear absolute memory
0xC4 CLRx clear indexed memory
0xD0 INCA increment accumulator A, affects ZNO
0xD1 INCB increment accumulator B
0xD2 INCa increment absolute address
0xD3 INCx increment indexed address
INCS #define #define INCS 0xD4 // increment PC Stack pointer
0xD5 INXS increment Auxiliary Stack pointer
0xD6 INCX increment Index Register
0xD7 DECA decrement accumulator A
0xD8 DECB decrement accumulator B
0xD9 DECa decrement absolute address
0xDA DECx decrement indexed address
DECS #define #define DECS 0xDB // decrement PC Stack pointer
0xDC DEXS decrement Auxiliary Stack pointer
0xDD DECX decrement Index Register
0xE0 USE capture hardware
0xE1 UNUSE release hardware
0xF0 TEMPO set tempo
0xF1 REST musical rest
0xF2 TONE play a tone, immediate
0xF3 TONEAB play a tone, values from accumulators
0xF4 TONEx play a tone, value from address in index reg
0xF7 RESET hard reset
0xF8 DUMP send register contents to serial
0xF9 TEST run test routine
0xFA RND load accumulators A & B with random values
0xFB PAUSE wait for keypress
0xFC DEBUG sets/reset debugOn
0xFD OK display ok
0xFE ERR display err
0xFF HALT terminates