DATA MANIPULATION:

```
Mov  dst,src
80   Mov  R8,R8              1000 0000
88   Mov  R16,R16            1000 1000
81   Mov  R8,$HH             1000 0001
89   Mov  R16,$HHHH          1000 1001
82   Mov  R8,[$MMMM]         1000 0010
8A   Mov  R16,[$MMMM]        1000 1010
83   Mov  [$MMMM],R8         1000 0011
8B   Mov  [$MMMM],R16        1000 1011
84   Mov  R8,[R16]           1000 0100
8C   Mov  R16,[R16]          1000 1100
85   Mov  [R16],R8           1000 0101
8D   Mov  [R16],R16          1000 1101
86   Mov  R8,[$MMMM,R8]      1000 0110
8E   Mov  R16,[$MMMM,R8]     1000 1110
87   Mov  [$MMMM,R8],R8      1000 0111
8F   Mov  [$MMMM,R8],R16     1000 1111

Lea  dst,src
*dst MUST be a 16-bit register
     LEA  R16,[$MMMM,R8]
```

MATH:

Addc dst,src

```
10   Addc R8,R8
11   Addc R8,$HH
12   Addc R8,[$MMMM]
13   Addc [$MMMM],R8
14   Addc R8,[R16]
15   Addc [R16],R8
16   Addc R8,[$MMMM,R8]
17   Addc [$MMMM,R8],R8
```

Subb

```
20   Subb R8,R8
21   Subb R8,$HH
22   Subb R8,[$MMMM]
23   Subb [$MMMM],R8
24   Subb R8,[R16]
25   Subb [R16], R8
26   Subb R8,[$MMMM,R8]
27   Subb [$MMMM,R8], R8
```

Cmp

```
30   Cmp  R8,R8
31   Cmp  R8,$HH
32   Cmp  R8,[$MMMM]
33   Cmp  [$MMMM],R8
34   Cmp  R8,[R16]
35   Cmp  [R16], R8
36   Cmp  R8,[$MMMM,R8]
37   Cmp  [$MMMM,R8], R8
```

LOGICAL:

Not

```
40   Not  R8
43   Not  [$MMMM]
47   Not  [$MMMM,R8]
```

And

```
50   And  R8,R8
51   And  R8,$HH
52   And  R8,[$MMMM]
53   And  [$MMMM],R8
54   And  R8,[R16]
55   And  [R16], R8
56   And  R8,[$MMMM,R8]
57   And  [$MMMM,R8], R8
```

Or

```
60   Or   R8,R8
61   Or   R8,$HH
62   Or   R8,[$MMMM]
63   Or   [$MMMM],R8
64   Or   R8,[R16]
65   Or   [R16], R8
66   Or   R8,[$MMMM,R8]
67   Or   [$MMMM,R8], R8
```

Xor

```
70   Xor  R8,R8
71   Xor  R8,$HH
72   Xor  R8,[$MMMM]
73   Xor  [$MMMM],R8
74   Xor  R8,[R16]
75   Xor  [R16], R8
76   Xor  R8,[$MMMM,R8]
77   Xor  [$MMMM,R8], R8
```

STACK:

Push

```
90   Push R8
98   Push R16
```

Pop

```
A0   Pop  R8
A8   Pop  R16
```

CONTROL:

Unconditional:

Jmp

```
B8   jmp  R16
BD   jmp  [R16]
B9   jmp  $MMMM
BB   jmp  [$MMMM]
BF   jmp  [$MMMM,R8]
```

Call

```
C8   Call R16
CD   Call [R16]
C9   Call $MMMM
CB   Call [$MMMM]
CF   Call [$MMMM,R8]
```

Ret

```
C0   No arguments
```

CONTROL:

Conditional:

All conditional jumps have only one addressing mode as follows:

Jcc <16-bit relative PC offset> 3 bytes total

Signed:

```
D0   Jgt        (N ^ V) = 0 && Z = 0  □ Z or (N ^ V) == 0 □
D1   Jge             (N ^ V) = 0
D2   Jlt             (N ^ V) = 1
D3   Jle             (N ^ V) = 1 || Z = 1
```

Unsigned:

```
D4   Jhi             C = 0 && Z = 0
D5   Jls             C = 1 || Z = 1
D6   Jlo  (jcs)      C = 1
D7   Jhs  (jcc)      C = 0
D8   Jeq  (jzs)      Z = 1
D9   Jne  (jzc)      Z = 0
DA   Jmi  (jss)      S = 1
DB   Jpl  (jsc)      S = 0
DC   Jvs             V = 1
DD   Jvc             V = 0
```

MISCELLANEOUS:

E0 Nop

E1 Swi

E2 Rti