Instruction Translation

MC → ASM

Disassembly of MC $\longrightarrow$ ASM

- write bit pattern in binary
- use bits 15-12 to identify the operation category
- use remaining bits to disambiguate, if necessary in the case of multiple instructions using the same 15-12
  - check the hardcoded bits for mismatches
  - check for invalid addressing modes
- you can check your work in Devpac

Example Disassemble $C747_{16}$

Express in Binary:

1100 0111 0100 0111

From the Operation Code Map:

AND	1100 DDD1 SSMM MRRR	**OR**	1100 DDD0 SSMM MRRR
MULS	1100 DDD0 11mm mrrr		
MULU	1100 DDD1 11mm mrrr		
ABCD	1100 RRR1 0000 0rrr	**OR**	1100 RRR1 0000 1rrr
EXG	1100 DDD1 0100 0DDD		
	1100 AAA1 0100 1AAA		
	1100 DDD1 1000 1AAA		

Example Disassemble $C747_{16}$

Express in Binary:

    1100 0111 0100 0111

Check the hardcoded bits for mismatches to eliminate possibilities:

```
AND     1100 DDD1 SSMM MRRR     OR   1100 DDD0 SSMM MRRR
MULS1100 DDD0 11mm mrrr
MULU1100 DDD1 11mm mrrr
ABCD1100 RRR1 0000 0rrr     OR   1100 RRR1 0000 1rrr
EXG     1100 DDD1 0100 0DDD
        1100 AAA1 0100 1AAA
        1100 DDD1 1000 1AAA
```

This leaves us two possibilities for the instruction

Example Disassemble $C747_{16}$

The pattern of: `1100 DDD1 SSEE EEEE`

Which is: `AND.x Dn,?`

The EE EEEE corresponds to MM MRRR, because **Dn** is in the source location.

So matching each component results in:

```
DDD =   111     7
SS  =   01      w
MMM =   000     Dn
```

check if valid dest

NOT allowed → so can't be this instruction

Example Disassemble $C747_{16}$

Therefore, the ONLY other possibility is:

```
    1100 DDD1 0100 0DDD
```

which is: `EXG Dn,Dn`

matching the components is

```
1st DDD =   011     3   src
2nd DDD =   111     7   dst
```

Therefore,the opcode is: `exg.l d3,d7`

Disassemble Exercise

1. `DD8B`
2. `BF4D`
3. `B3C0`
4. `7408`
5. `FFFF`

Disassemble Exercise - Solutions

```
1.  DD8B→   addx.l  -(a3),-(a6)

2.  BF4D→   cmpm.w  (a5)+,(sp)+

3.  B3C0→   cmpa.l  d0,a1

4.  7408→   moveq.b #8,d2

5.  FFFF→   dc.w$FFFF   illegal opcode
```