Patches to ASM 2.0

The following patches fix two bugs in the Digital Research ASM v2.0 assembler. The patches bring ASM to version 2.1. Note, this a 2018 patch by Mike Douglas, not an original DRI patch.

SET directive bug

The very first time each symbol associated with a SET directive is entered into the symbol table (i.e., when first encountered during pass 1 of the assembler), the value assigned to the symbol is zero instead of the specified value. All subsequent encounters of that symbol assign the proper value. If a symbol is assigned only once prior to a conditional assembly statement that references that symbol, then the symbol could have a different value on the first pass than during the second pass. For example, an intended "true" conditional assembly will not generate code on the first pass because the symbol is erroneously assigned a zero value, and will generate code on the second pass, resulting in a phase error.

When any symbol is first entered into the symbol table, the "type" field is set to zero (undefined). Subsequent code then sets the type field. However, for the SET directive, a check is made to verify the type is "SET" prior to the type field being updated. Therefore, this first compare always fails and the code that evaluates the symbol value expression is bypassed – leaving the symbol value zero.

The patch below fixes this bug by eliminating the effect of the failed type field comparison in the SET code. This is done by disabling the call to the label error routine. The CNZ opcode is replaced with a LDA opcode which effectively NOP's the call.

```
1DB2h from C4  (CNZ)
        to 3A  (LDA)
```

IF directive bug

Regarding evaluation of the expression following the IF directive, the ASM manual states "If the expression evaluates to a nonzero value, then statement #1 through statement #n are assembled. If the expression evaluates to zero, the statements are listed but not assembled."

However, this is not the case. The expression is considered TRUE if the least significant bit of the expression is one and FALSE if the least significant bit of the expression is zero. The upper 15 bits of the expression are ignored.

The patch below fixes this bug by doing a full 16 bit test for zero instead of testing bit 0 only.

```
1D4F from 1F DA  (RAR, JC)
       to B4 C2  (ORA H, JNZ)
```

Version Number Patch

The patch below updates the version number from 2.0 to 2.1

```
0FB7h from 30   ("0")
        to 31   ("1")
```