1 Introduction

2 Teamwork planning

3 Syntax in EBNF language

```
file =   'DEVICES', {DEV, ','}, DEV, ';', 'CONNECT', {CON, ','}, CON, ';',
         'MONITOR', {MON, ','}, MON, ';';
DEV  =   'CLOCK', DEV_NAME, digit, {digit}                              |
         'SWITCH', DEV_NAME, ( 1 | 0 )                                  |
         'AND' | 'NAND' | 'OR' | 'NOR', DEV_NAME, [1], digit            |
         'D_TYPE', DEV_NAME                                             |
         'XOR', DEV_NAME;
DEV_NAME* =      digit | letter, {digit | letter | '_'};
CON       =      O_PIN, '=>', I_PIN;
O_PIN     =      DEV_NAME                        |
                 DEV_NAME, '.', 'Q' | 'QBAR';
I_PIN     =      DEV_NAME, '.', 'I', [1], digit          |
                 DEV_NAME, '.', 'DATA'|'CLK'|'SET'|'CLEAR';
MON       =      O_PIN | I_PIN;
```

*DEV_NAME = [0−9a−zA−Z_]+, DEV_NAME can be any combination of letter and number and '_', other than "DEVICES", "CONNECT", "MONITOR", "CLOCK", "SWITCH", "AND", "NAND', "OR", "NOR", "D_TYPE", "XOR"

4 Syntax error identification and handling

5 Semantics error identification and handling

6 Example definition files

Circuit 1 definition file.

```
DEVICES AND A 1,
        OR  B 1,
        XOR C,
        NAND D 3,
        SWITCH S1 1,
        SWITCH S2 1,
        SWITCH S3 1,
        SWITCH S4 1;
CONNECT S1 => A.I1,
        S2 => B.I1,
        S3 => C.I1,
        S4 => C.I2,
        A  => D.I1,
        B  => D.I2,
```

```
        C   => D.I3;
MONITOR D;
```

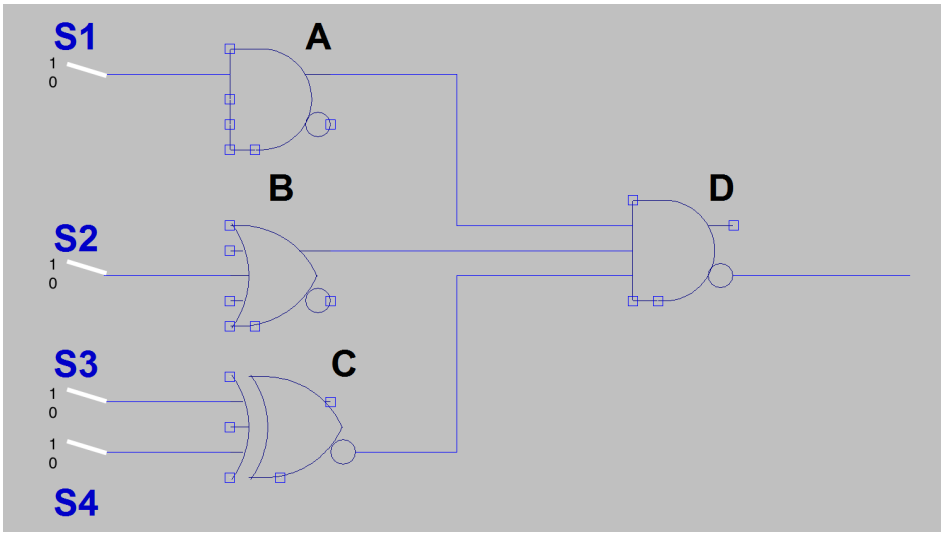

Figure 1: Circuit1

Circuit 2 definition file.

```
DEVICES CLOCK L 100,
        SWITCH S1 1,
        SWITCH S2 0,
        SWITCH S3 0,
        DTYPE M,
        NOR A 2;
CONNECT S1 => M.SET,
        S2 => M.DATA,
        S3 => M.CLEAR,
        L  => M.CLK,
        M.Q => A.I1,
        M.QBAR => A.I2;
MONITOR A,
        QBAR;
```

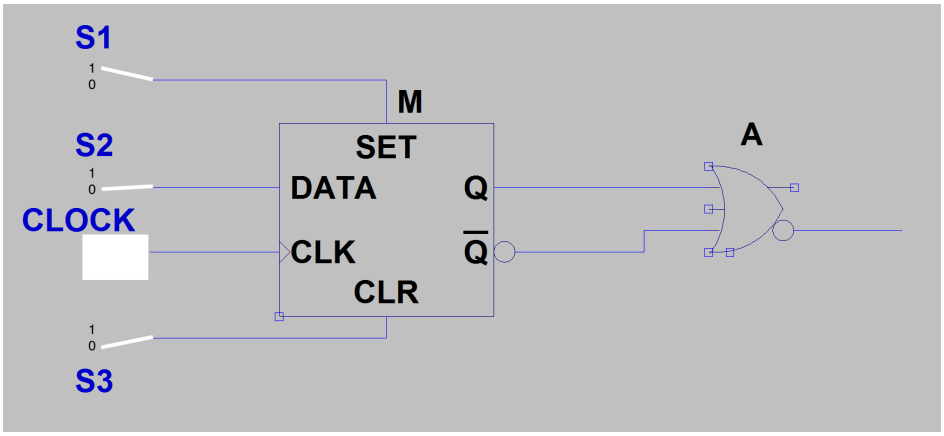

Figure 2: Circuit1