|  |
| --- |
| HES-SO |
| Projet C++ |
| Protocole JSON |

|  |
| --- |
| Gilles Mottiez  01/05/2018 |

Ce document décrit l’implémentation des fichier .json qui représentent un circuit de logique numérique :

1. Chaque fichier commence avec l’attribut « name », puis de l’attribut « nGates » qui définit le nombre total de portes logiques du circuit.
2. Vient ensuite un tableau de portes logiques « gates », qui répertorie toutes les portes du circuit
3. Une porte logique est définie comme telle :
   1. Attribut « type » : définit la nature même de la porte
   2. Attribut « ID » : chaque porte logique possède un ID différent qui permet
   3. Attribut « level » : définit l’enchainement logique des portes :
      1. Une porte de level 0 possède les entrées du système
      2. Une porte de level maximum possède la ou les sorties du système
   4. Attribut « nInput » qui spécifie le nombre d’entrées de la porte
   5. Tableau « pins » : contient les connecteurs de la porte :
      1. Chaque connecteur a un attribut label tel que :
         1. « ID » +
         2. Numéro du connecteur
         3. Un label de pin input 1 pour l’ID O1 est : « O1\_I1 »
      2. Un attribut « connected » qui contient le label du connecteur en face
4. Les attributs « type » sont définis tels que :
   1. AND
   2. OR
   3. NOT
5. Les attributs « level » sont définis tels que :
   1. Les blocs d’entrée du système on le level 0
   2. Les blocs de sortie du système ont le level le plus haut
   3. Chaque sortie d’un bloc incrémente d’1 le level du bloc d’entrée suivant
6. Les signaux logiques d’entrées sont dénommés comme suit : LOG\_LOW pour un 0 logique, LOG\_HIGH pour un 1 logique
7. Les signaux de sortie des portes logiques sont définis tel que :
   1. « ID »
   2. + « \_U »

Un bloc avec l’ID « O1 » possède un signal de sortie de label « O1\_U »

Exemple d’implémentation :

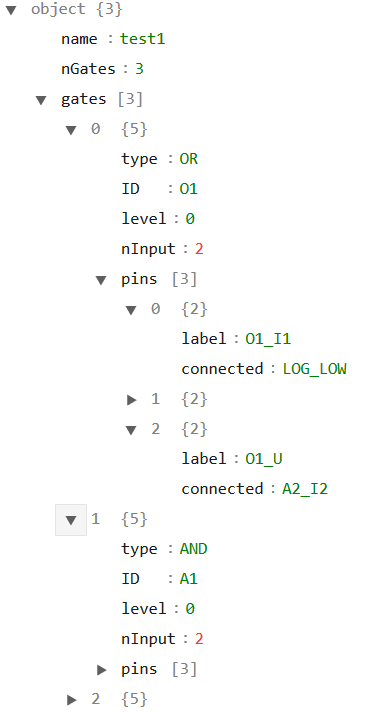


Figure 1 : code JSON

Est la représentation JSON du circuit logique suivant :

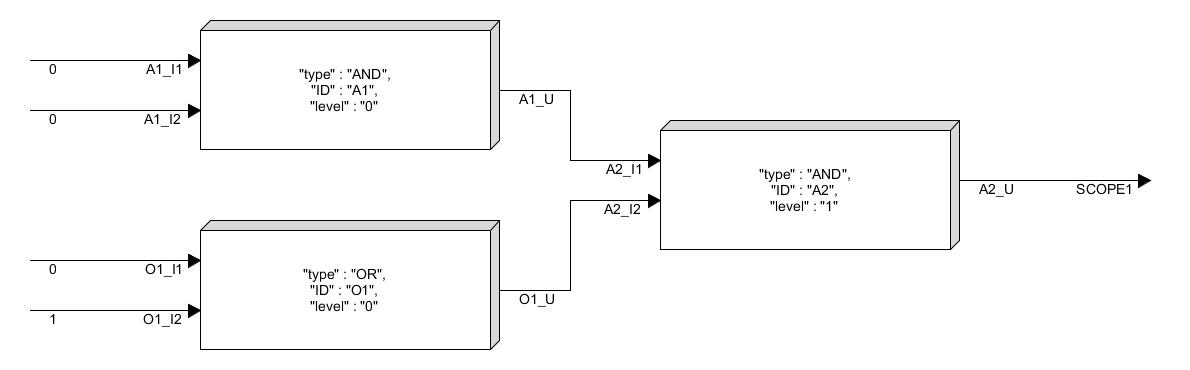


Figure 2 : circuit logique