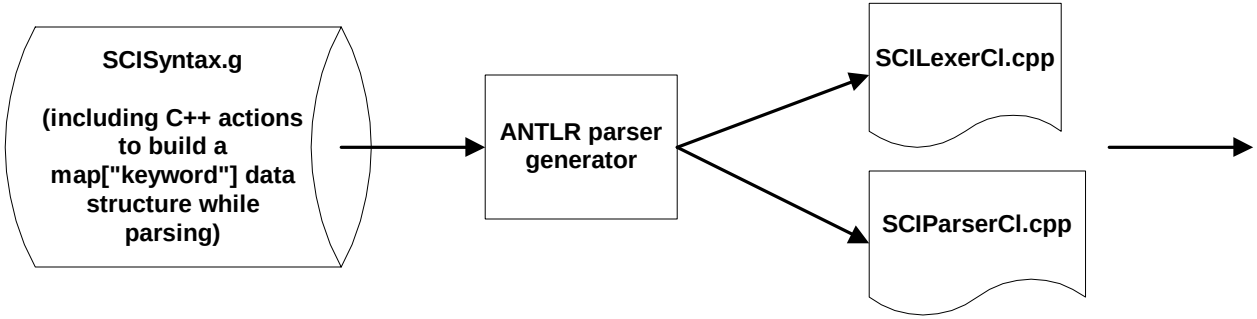
Proposal for Model Configuration:

(1) Need to specify the syntax of the model configuration file; and can use a C++ lexer/paser generator, such as ANTLR (http://www.antlr.org/), to automatically generate corresponding C++ parsing functions to be used by the OCP ParamCl class. The advantage is low maintenance cost.
(2) Configure an OCP channel by reading in a configuration file containing OCP parameters
(3) Access OCP parameters using the map["keyword"] container class provided by the C++ standard template library.
(4) Master or slave cores can also have their own configuration; either using a global configuration file or multiple files.

Step 1:
- provide a grammar description for the syntax of the configuration file
- generate C++ lexer and parser code automatically
- apply the lexer/parser class/methods in the ocp_tl_param_cl.h file



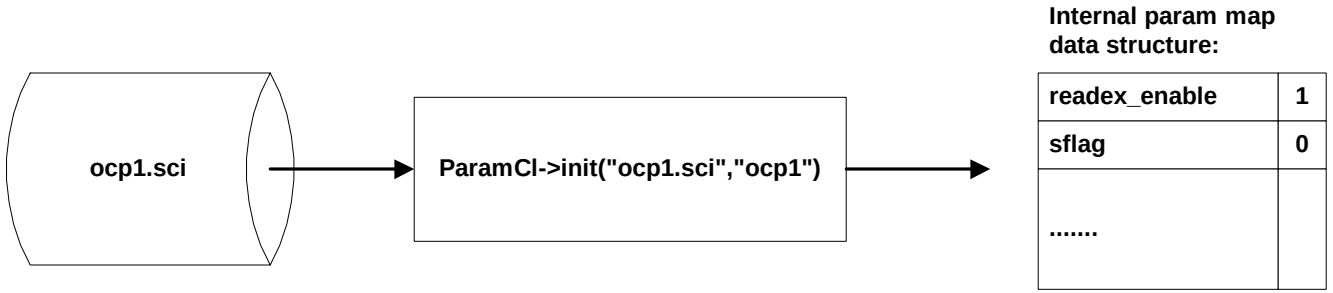
ocp_tl_param_cl.h (pseudo code)

```
class ParamCl {
   std::map<std::string,int> param;

   void
   ParamCl::init(string file_name,
                 string ocp_name)
   {
      inp = open(file_name);
      SCILexerCl lexer(inp);
      SCIParserCl parser(lexer);

      parser.program(ocp_name,
                     param);
   }
}
```

Step 2: internal map["keyword"] data structure is generated while parsing a configuration file



Internal param map data structure:

readex_enable	1
sflag	0
.....	

Step 3: use model

```
// while issuing requests in a master core
if (ParamCl->param["readex_enable"]) {
   // can generate read and readex requests
   .....
} else {
   // can only generate read requests
   .....
}
```