New CLI for DPDK

Keith Wiles, Intel

DPDK Summit - San Jose – 2017

#DPDKSummit

Why yet another CLI? ☹

- For many years I have been looking for a simple and easier to use API for development and production use, but I have not found one ☺
- Most of the Command Line interfaces I have seen are too complex or way too simple or use TCL, Python or some other language to create the interface.
  - These languages add a lot of code, plus are not super friendly for embedded apps in code size
  - Others are written in C++ or C, plus they try to do everything like convert strings to numbers and many other conversions (similar to DPDK's CommandLine).
- These conversions and complex structures are not required for many applications and add a lot of code to the interface.
- So not being able to find one that I liked, I decided to write my own

What makes a good CLI?

- A goal for a CLI is to create a quick and simple easy to use interface for Developers/Users
  - Allowing the developer to add a new command or debug must be quick and simple
- Use well known developer constructs to make learning the new interface simple
- A CLI should allow for dynamically adding and removing command at run time
- Be able to create complexed commands without complex structures
- Allow for hierarchical commands instead of a flat set of commands
- Make the user interface simple and familiar
- Must have autocomplete and history of commands to run or re-run them quickly
- Plus a number of other features

CLI Features

- CLI has no global variables, which allows for multiple or different user interfaces in the same process, e.g. restricted, power and admin users, …
- CLI support commands, files, aliases and directories
  - CLI is designed around a shell/directory like user interface
- Callbacks from commands/files use the simple argc/argv function interface
- Simple structures to add and remove commands, files, directories
- Simple environment variable support, plus help support
- For complex commands, we have the MAP interface to make it simpler
  - MAP is a set of ‘printf’ like strings to define commands and how they are parsed/found
- CLI uses a simple shell and directory format for commands/files, which gives the developer a hierarchy of commands

Testpmd application in DPDK

- The testpmd application in DPDK is used to test and debug DPDK and it has a LOT of commands
- I decide to convert Testpmd to use the new CLI to test out if it would work out better
  - Old cmdline.c file is about: 12K lines of code
  - New cli_cmds.c file is about: 4.5K lines of code
- It took about two days to add the new CLI commands to testpmd, without doing a lot of testing ☺
- What reduced the line count was removing cmdline structures and reducing the number of functions (which were required for each command line and variation of command lines)
  - Converting these complex command lines to use CLI's MAP style interface

Simple Example of CLI code

- First task is to initialize the command line interface
- Taking just the defaults makes a command line easy
- The cli_start(const char *msg)
  - If msg is not NULL then print the string to the console on startup
- When the user types ‘quit’ or control-X cli_start() will return
- Gives some basic commands like ls, pwd, more, env, echo, history, …

```
#include <cli.h>

int main(int argc, char **argv)
{
    if (!cli_create_with_defaults()) {

        cli_start(NULL);

        cli_destroy();
    }
    return 0;
}
```

Example 2 using a tree

DPDK

```
#include <cli.h>

static int hi_cmd(int argc, char **argv)
{
    cli_printf("Hello World!, - %s\n",
        (argc > 1)? argv[1] : "y'all");
    return 0;
}

static struct cli_tree my_tree [] = {
    c_dir("/bin"),
    c_cmd("hi",  hi_cmd, "Hello World"),
    c_end()
};
```

```
static int mytree(void)
{
    if (cli_default_tree_init())
        return -1;
    if (cli_add_tree(NULL, my_tree))
        return -1;
    return cli_add_bin_path("/bin");
}

int main(int argc, char **argv)
{
    if (!cli_create_with_tree(mytree)) {
        cli_start(NULL);
        cli_destroy();
    }
    return 0;
}
```

New CLI Summary

- CLI– A Command Line Interface
  - Source: http://dpdk.org/browse/draft/dpdk-draft-cli ‘cli’ branch
- Examples are at: http://dpdk.org/browse/apps/pktgen-dpdk Also in the source above
- In the DPDK/lib/librte_cli directory are two *.rst files and README file for more documentation of CLI
- The PKTGEN application is now converted to use the CLI interface

DEMO

DPDK

- A Simple CLI Demo running in a VM