ABC: A System for Sequential Logic Synthesis and Formal Verification

Yung-Chih Chen

Some materials were adapted from

- Alan Mishchenko, “ABC: An Industrial-Strength Logic Synthesis and Verification Tool”
- Ana Petkovska, “Getting started with ABC”
- Logic Synthesis & Verification @ NTU

Outline

- Introduction to ABC
- Using ABC
- Programming ABC
- Program Assignment 1

What is ABC?

- A powerful academic tool for logic synthesis and verification
  - Developed by Berkeley Logic Synthesis and Verification Group
  - Fast and scalable logic optimization based on And-Inverter-Graph (AIG)
  - Optimal-delay DAG-based technology mapping for look-up tables and standard cells
  - Innovative algorithms for sequential synthesis and verification

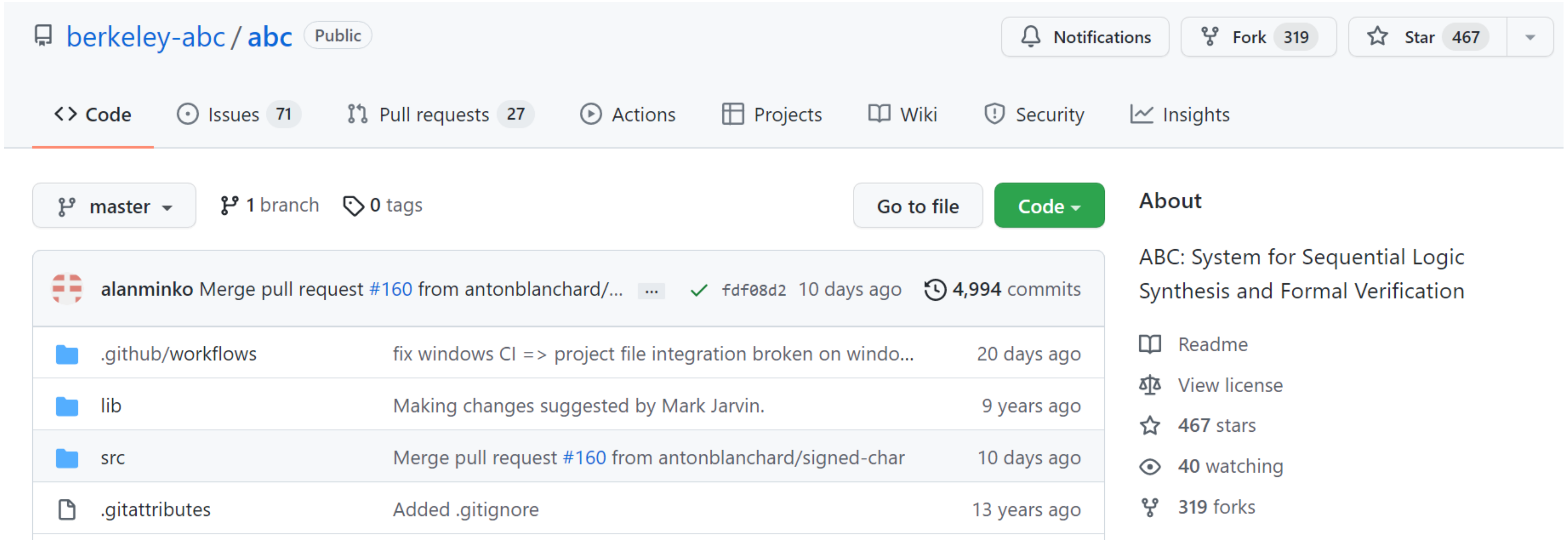
- Programming environment
  - Open-source
    - You can also customize ABC for your needs
  - Evolving and improving over time

ABC Resources

- Latest ABC code can be found at https://github.com/berkeley-abc/abc
- "Getting started with ABC", a tutorial by Ana Petkovska
  - https://www.dropbox.com/s/qrl9svlf0ylxy8p/ABC_GettingStarted.pdf
- An overview paper
  - R. Brayton and A. Mishchenko, "ABC: An academic industrial-strength verification tool", Proc. CAV'10
- Website: https://people.eecs.berkeley.edu/~alanmi/abc/
  - Command summary, programming notes, …

Using ABC

- Download ABC
  - https://github.com/berkeley-abc/abc



berkeley-abc / abc Public

Notifications | Fork 319 | Star 467

Code | Issues 71 | Pull requests 27 | Actions | Projects | Wiki | Security | Insights

master | 1 branch | 0 tags | Go to file | Code

alanminko Merge pull request #160 from antonblanchard/... ... ✓ fdf08d2 10 days ago 4,994 commits

.github/workflows	fix windows CI => project file integration broken on windo...	20 days ago
lib	Making changes suggested by Mark Jarvin.	9 years ago
src	Merge pull request #160 from antonblanchard/signed-char	10 days ago
.gitattributes	Added .gitignore	13 years ago

About

ABC: System for Sequential Logic Synthesis and Formal Verification

Readme

View license

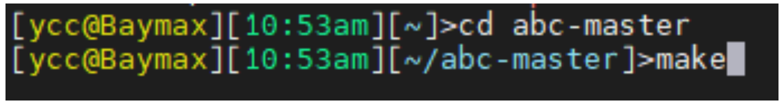
467 stars

40 watching

319 forks

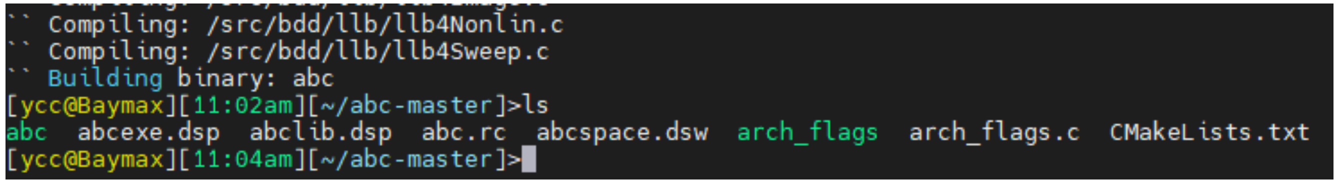
Install ABC

- Download and unzip the code, and go into the directory
  - Compile ABC as a binary (execution file, stand-alone mode)
    - Type make



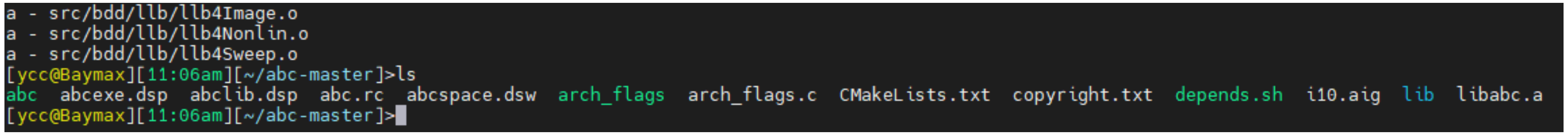
```
[ycc@Baymax][10:53am][~]>cd abc-master
[ycc@Baymax][10:53am][~/abc-master]>make
```

    - If the process ends successfully, you get



```
`` Compiling: /src/bdd/llb/llb4Nonlin.c
`` Compiling: /src/bdd/llb/llb4Sweep.c
`` Building binary: abc
[ycc@Baymax][11:02am][~/abc-master]>ls
abc  abcexe.dsp  abclib.dsp  abc.rc  abcspace.dsw  arch_flags  arch_flags.c  CMakeLists.txt
[ycc@Baymax][11:04am][~/abc-master]>
```

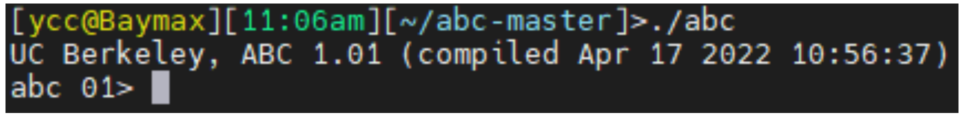
  - Compile ABC as a static library (API mode)
    - Type make libabc.a
    - Again, you get



```
a - src/bdd/llb/llb4Image.o
a - src/bdd/llb/llb4Nonlin.o
a - src/bdd/llb/llb4Sweep.o
[ycc@Baymax][11:06am][~/abc-master]>ls
abc  abcexe.dsp  abclib.dsp  abc.rc  abcspace.dsw  arch_flags  arch_flags.c  CMakeLists.txt  copyright.txt  depends.sh  i10.aig  lib  libabc.a
[ycc@Baymax][11:06am][~/abc-master]>
```

Run ABC in Stand-Alone Mode

- Type ./abc

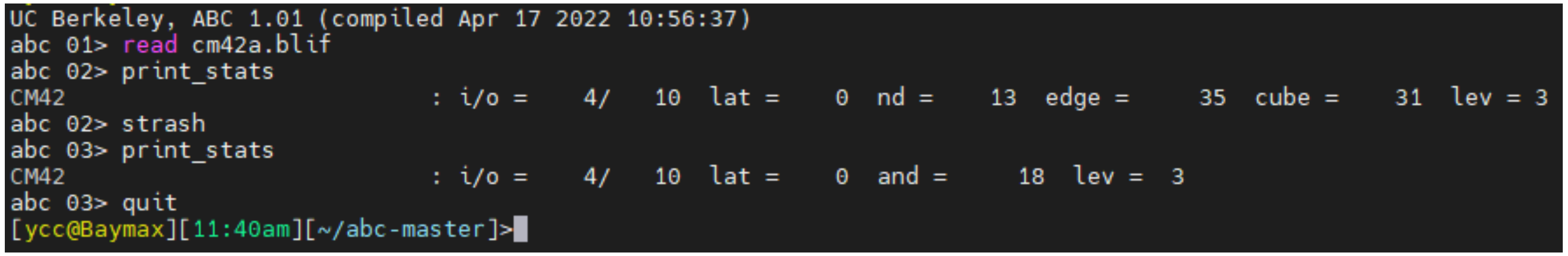


```
[ycc@Baymax][11:06am][~/abc-master]>./abc
UC Berkeley, ABC 1.01 (compiled Apr 17 2022 10:56:37)
abc 01>
```

Where you can execute commands implemented into ABC

- Type help to see all the supported commands

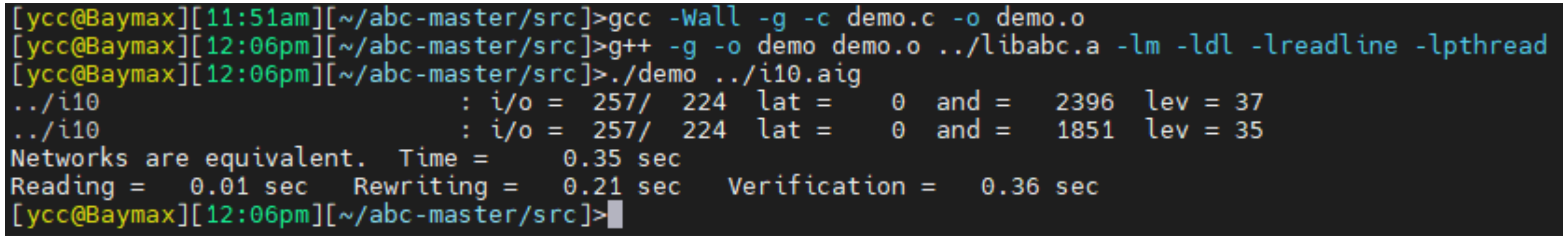
- Example
  - Copy the released cm42.blif to the directory



```
UC Berkeley, ABC 1.01 (compiled Apr 17 2022 10:56:37)
abc 01> read cm42a.blif
abc 02> print_stats
CM42                          : i/o =    4/   10  lat =    0  nd =    13  edge =     35  cube =    31  lev = 3
abc 02> strash
abc 03> print_stats
CM42                          : i/o =    4/   10  lat =    0  and =     18  lev =  3
abc 03> quit
[ycc@Baymax][11:40am][~/abc-master]>
```

Run ABC in the API Mode

- First, make sure libabc.a is ready
- Follow the instructions in README.md to compile and run demo.c (in src/)



```
[ycc@Baymax][11:51am][~/abc-master/src]>gcc -Wall -g -c demo.c -o demo.o
[ycc@Baymax][12:06pm][~/abc-master/src]>g++ -g -o demo demo.o ../libabc.a -lm -ldl -lreadline -lpthread
[ycc@Baymax][12:06pm][~/abc-master/src]>./demo ../i10.aig
../i10                        : i/o =  257/  224  lat =    0  and =   2396  lev = 37
../i10                        : i/o =  257/  224  lat =    0  and =   1851  lev = 35
Networks are equivalent.  Time =     0.35 sec
Reading =    0.01 sec   Rewriting =    0.21 sec   Verification =   0.36 sec
[ycc@Baymax][12:06pm][~/abc-master/src]>
```

  - You can see that, in demo.c, Cmd_CommandExecute(pAbc, Command) is used to call the commands in ABC

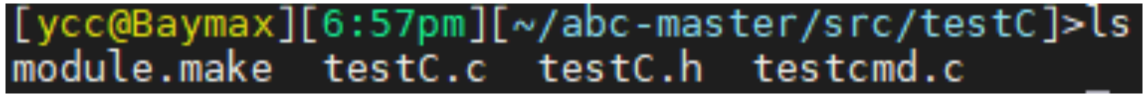
Inside ABC

- Most of the implemented commands are defined in the following files
  - src/base/abci/abc.c
  - src/base/io/io.c
- The declarations of the basic commands for working with ABC networks can be found in
  - src/base/abc/abc.h

Programming ABC

- Create a new command (your own) into ABC

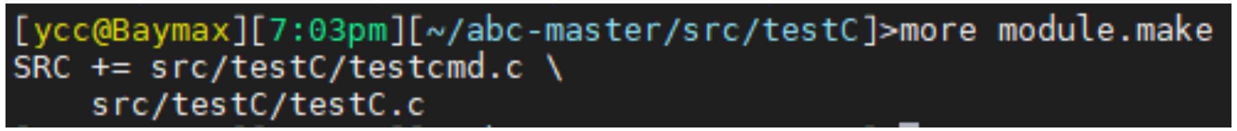
- In the fold src/, create a new folder testC/ with the following files
  - module.make, where you will list your .c files for compilation
  - testcmd.c, where you will declare and define your commands
  - testC.c, where you will define your main functions
  - testC.h, where you will declare your main functions



```
[ycc@Baymax][6:57pm][~/abc-master/src/testC]>ls
module.make   testC.c   testC.h   testcmd.c
```

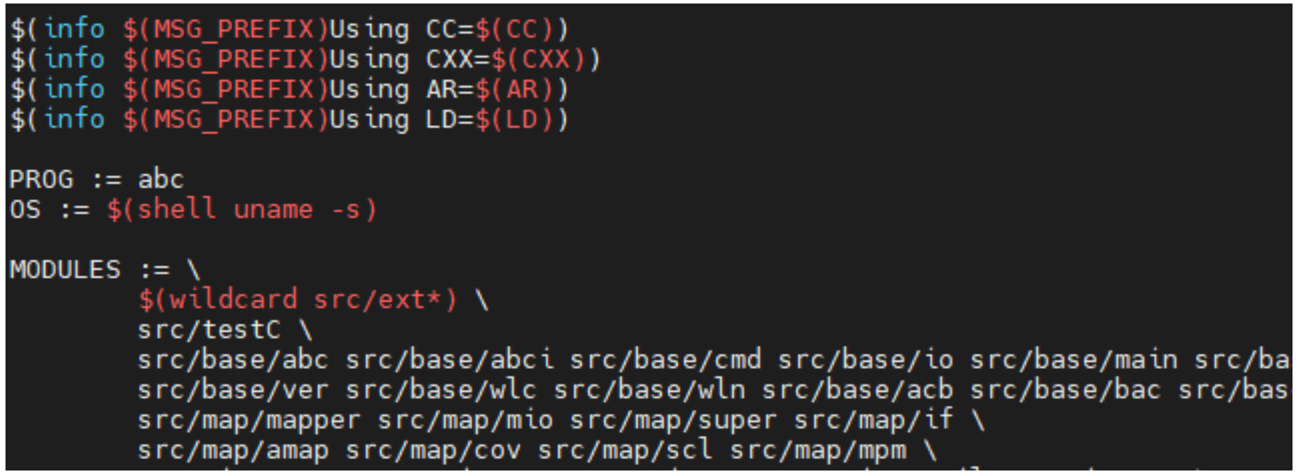
File: module.make

- List your .c files for compilation



```
[ycc@Baymax][7:03pm][~/abc-master/src/testC]>more module.make
SRC += src/testC/testcmd.c \
    src/testC/testC.c
```

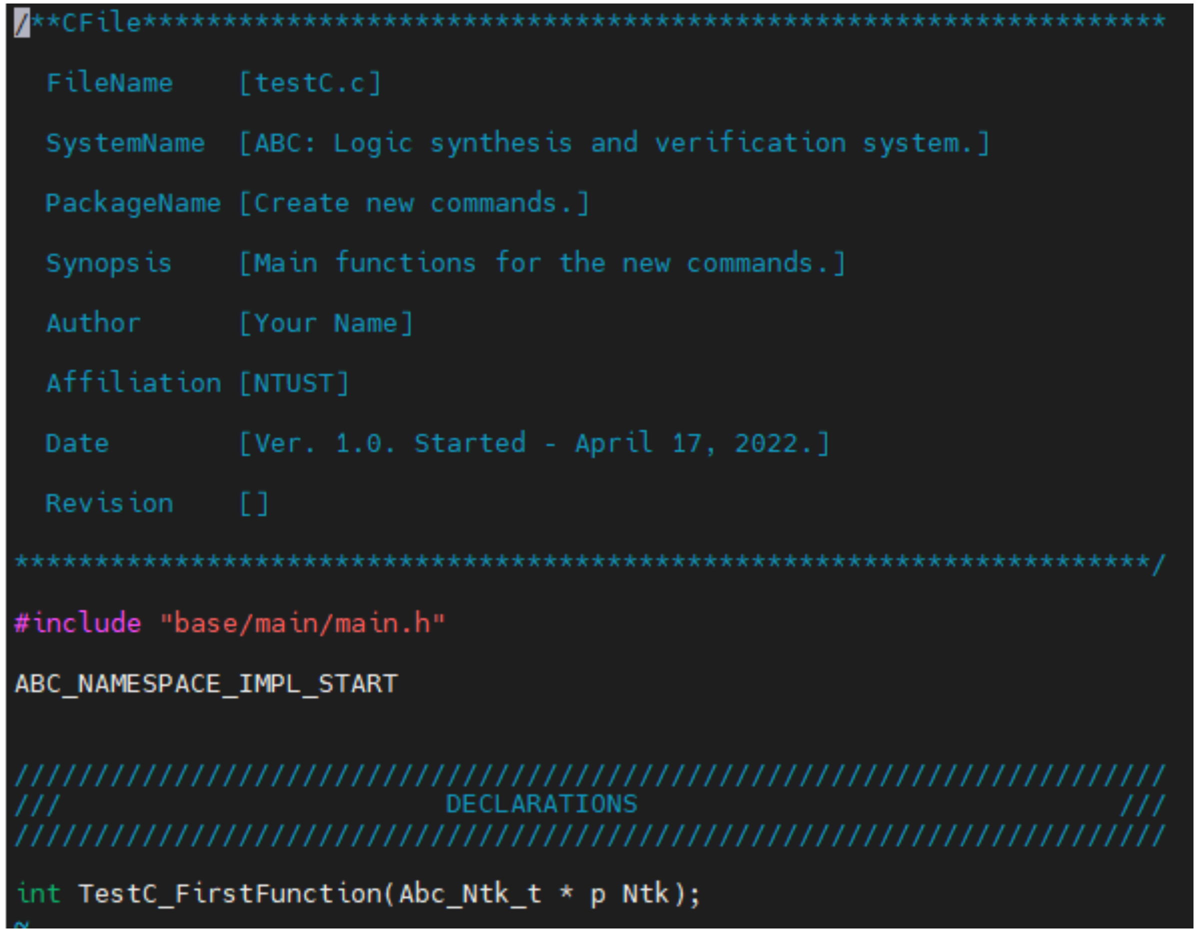
- You also need to list the folder as a new module in the Makefile [abc-master/Makefile]



```
$(info $(MSG_PREFIX)Using CC=$(CC))
$(info $(MSG_PREFIX)Using CXX=$(CXX))
$(info $(MSG_PREFIX)Using AR=$(AR))
$(info $(MSG_PREFIX)Using LD=$(LD))

PROG := abc
OS := $(shell uname -s)

MODULES := \
        $(wildcard src/ext*) \
        src/testC \
        src/base/abc src/base/abci src/base/cmd src/base/io src/base/main src/ba
        src/base/ver src/base/wlc src/base/wln src/base/acb src/base/bac src/bas
        src/map/mapper src/map/mio src/map/super src/map/if \
        src/map/amap src/map/cov src/map/scl src/map/mpm \
```

File: testC.c [1/3]

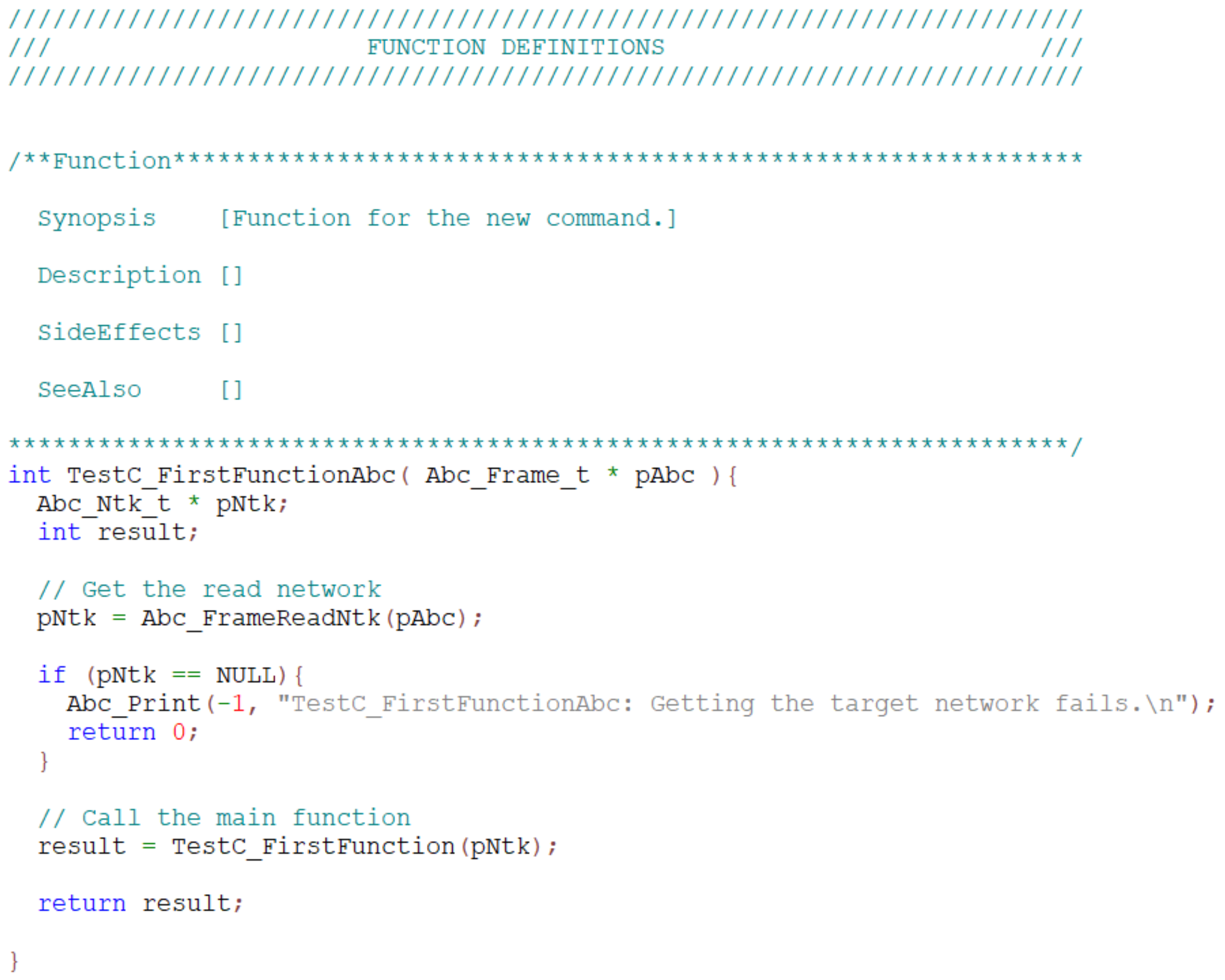
- Start with information about the file
- List needed libraries and the declarations of the functions



```
/**CFile****************************************************************

  FileName    [testC.c]

  SystemName  [ABC: Logic synthesis and verification system.]

  PackageName [Create new commands.]

  Synopsis    [Main functions for the new commands.]

  Author      [Your Name]

  Affiliation [NTUST]

  Date        [Ver. 1.0. Started - April 17, 2022.]

  Revision    []

***********************************************************************/

#include "base/main/main.h"

ABC_NAMESPACE_IMPL_START


////////////////////////////////////////////////////////////////////////
///                        DECLARATIONS                              ///
////////////////////////////////////////////////////////////////////////

int TestC_FirstFunction(Abc_Ntk_t * p Ntk);
```

File: testC.c [2/3]

- Define the function to be called by the new command
  - It extracts the network that is read into ABC and calls another function



```
////////////////////////////////////////////////////////////////////////
///                       FUNCTION DEFINITIONS                       ///
////////////////////////////////////////////////////////////////////////


/**Function*************************************************************

  Synopsis    [Function for the new command.]

  Description []

  SideEffects []

  SeeAlso     []

***********************************************************************/
int TestC_FirstFunctionAbc( Abc_Frame_t * pAbc ){
  Abc_Ntk_t * pNtk;
  int result;

  // Get the read network
  pNtk = Abc_FrameReadNtk(pAbc);

  if (pNtk == NULL){
    Abc_Print(-1, "TestC_FirstFunctionAbc: Getting the target network fails.\n");
    return 0;
  }

  // Call the main function
  result = TestC_FirstFunction(pNtk);

  return result;

}
```

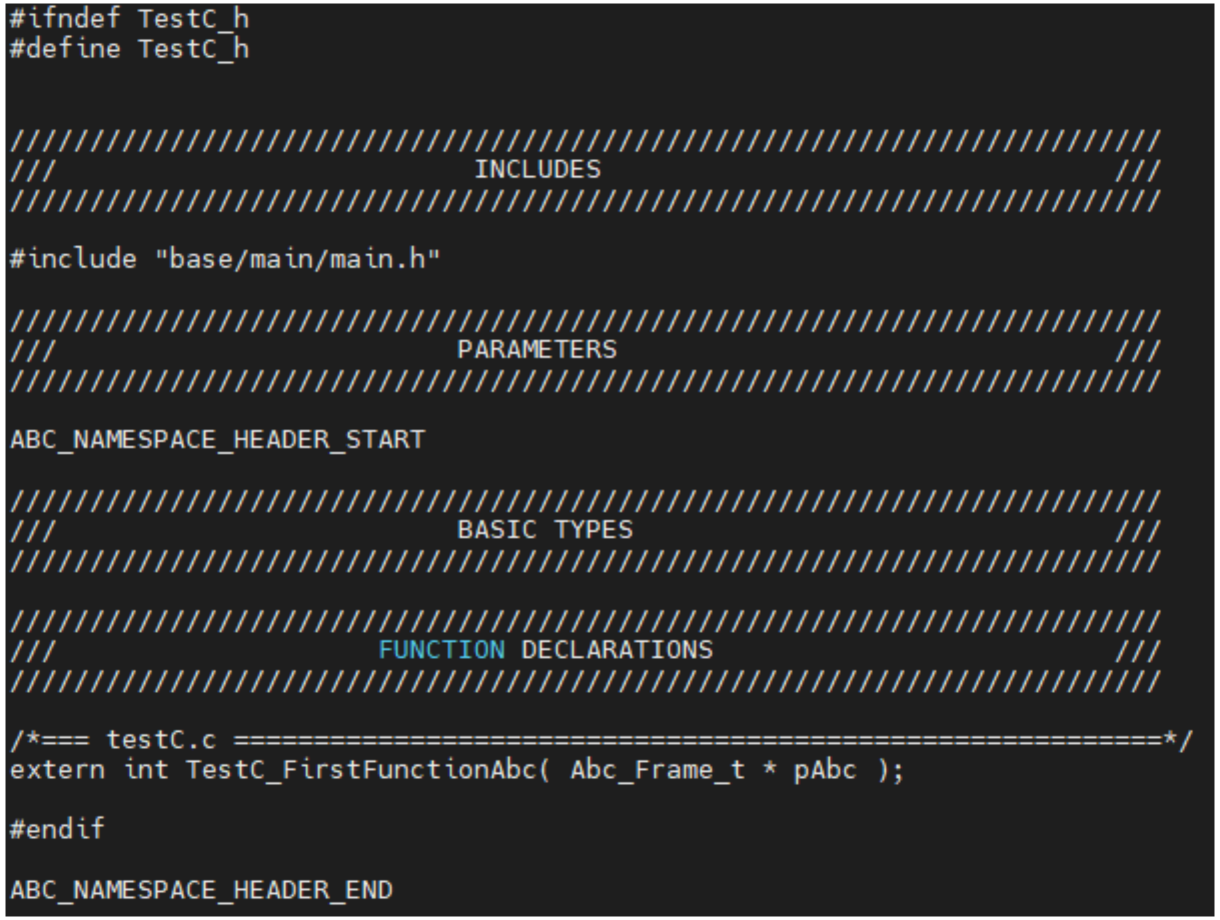
File: testC.c [3/3]

- Define a function to print information of the read network

```c
/**Function*************************************************************

  Synopsis    [Main function for the new command.]

  Description []

  SideEffects []

  SeeAlso     []

***********************************************************************/
int TestC_FirstFunction(Abc_Ntk_t * pNtk){
  // checked if the network is strashed
  if(!Abc_NtkIsStrash(pNtk)){
    Abc_Print(-1, "TestC_FirstFunction: This command is only applicable to strashed networks.\n");
    return 0;
  }

  // print information about the network;
  Abc_Print(1, "The network %s has:\n", Abc_NtkName(pNtk));
  Abc_Print(1, "\t- %d primary inputs;\n", Abc_NtkPiNum(pNtk));
  Abc_Print(1, "\t- %d primary outputs;\n", Abc_NtkPoNum(pNtk));
  Abc_Print(1, "\t- %d nodes;\n", Abc_NtkNodeNum(pNtk));
  return 1;
}

////////////////////////////////////////////////////////////////////////
///                       END OF FILE                                ///
////////////////////////////////////////////////////////////////////////


ABC_NAMESPACE_IMPL_END
```

File: testC.h

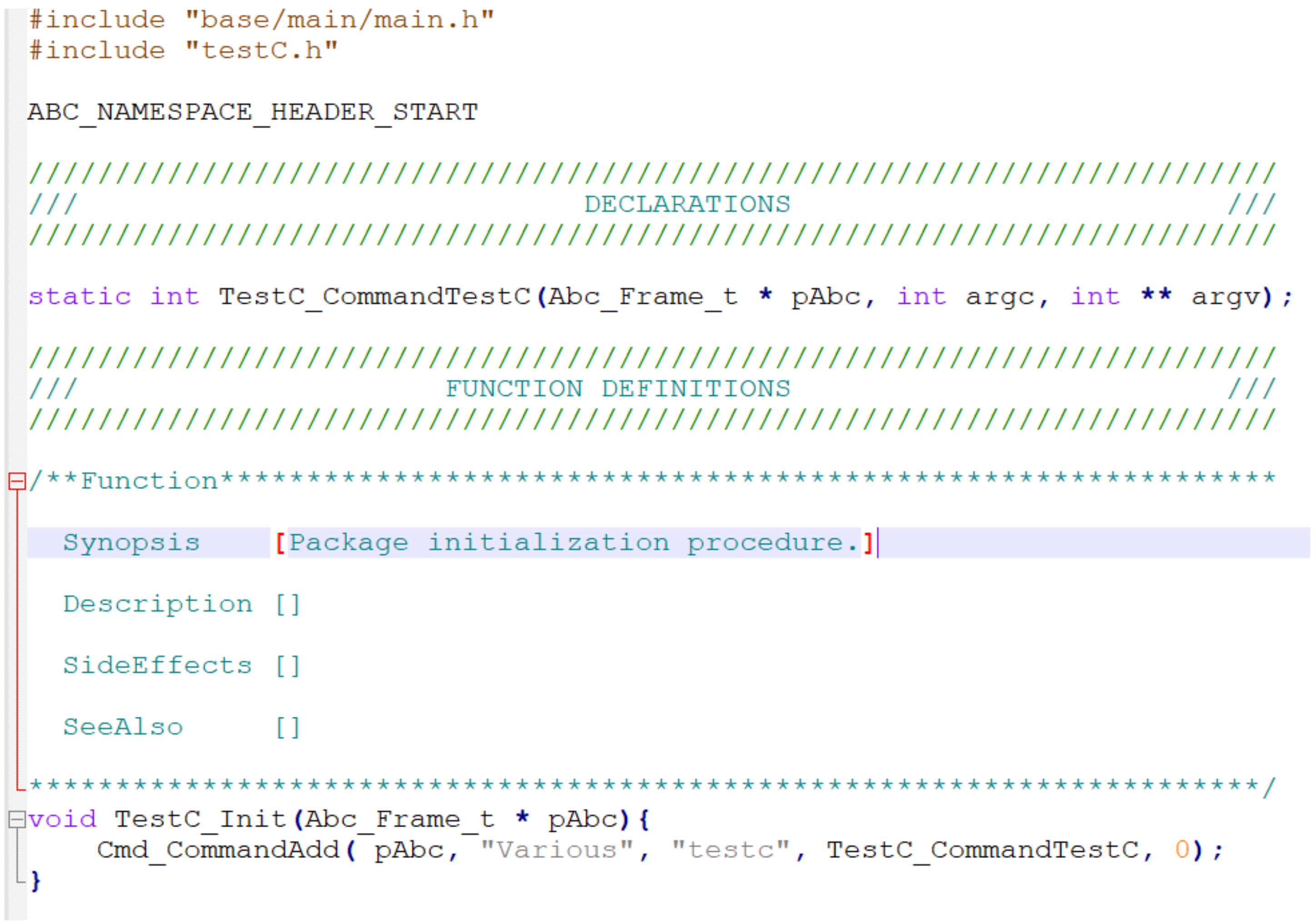
- Declare the functions that can be globally used



```
#ifndef TestC_h
#define TestC_h


////////////////////////////////////////////////////////////////////////
///                          INCLUDES                                ///
////////////////////////////////////////////////////////////////////////

#include "base/main/main.h"

////////////////////////////////////////////////////////////////////////
///                         PARAMETERS                               ///
////////////////////////////////////////////////////////////////////////

ABC_NAMESPACE_HEADER_START

////////////////////////////////////////////////////////////////////////
///                         BASIC TYPES                              ///
////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////
///                    FUNCTION DECLARATIONS                         ///
////////////////////////////////////////////////////////////////////////

/*=== testC.c  =========================================================*/
extern int TestC_FirstFunctionAbc( Abc_Frame_t * pAbc );

#endif

ABC_NAMESPACE_HEADER_END
```

File: testcmd.c (1/2)

- List needed libraries and declarations of the functions that define the commands
- Include one initialization function for module initialization and for inserting the command



```
#include "base/main/main.h"
#include "testC.h"

ABC_NAMESPACE_HEADER_START

////////////////////////////////////////////////////////////////////////
///                        DECLARATIONS                              ///
////////////////////////////////////////////////////////////////////////

static int TestC_CommandTestC(Abc_Frame_t * pAbc, int argc, int ** argv);

////////////////////////////////////////////////////////////////////////
///                     FUNCTION DEFINITIONS                         ///
////////////////////////////////////////////////////////////////////////

/**Function*************************************************************

  Synopsis    [Package initialization procedure.]

  Description []

  SideEffects []

  SeeAlso     []

***********************************************************************/
void TestC_Init(Abc_Frame_t * pAbc){
    Cmd_CommandAdd( pAbc, "Various", "testc", TestC_CommandTestC, 0);
}
```

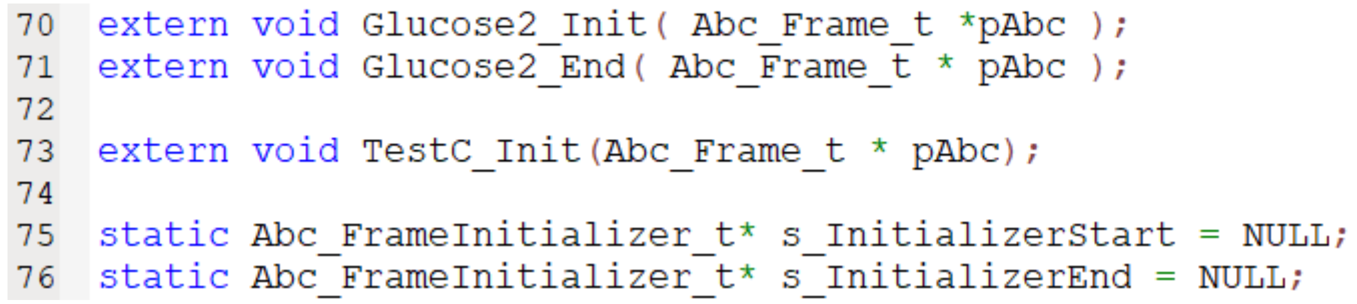
File: testcmd.c (2/2)

- **Give the definitions of the functions that implement our commands**

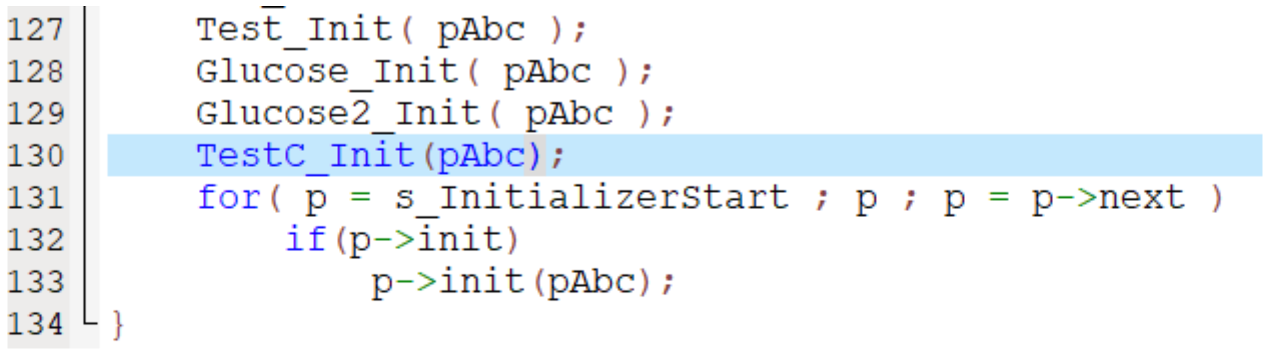
```
int TestC_CommandTestC(Abc_Frame_t * pAbc, int argc, int ** argv){
    int fVerbose;
    int c, result;
    fVerbose = 0;
    Extra_UtilGetoptReset();
    while ( ( c = Extra_UtilGetopt( argc, argv, "wh" ) ) != EOF )
    {
        switch ( c )
        {
        case 'v':
            fVerbose ^= 1;
            break;
        case 'h':
            goto usage;
        default:
            goto usage;
        }
    }

    result = TestC_FirstFunctionAbc( pAbc );

    if ( fVerbose )
    {
        Abc_Print( 1, "\nVerbose mode is on.\n" );
        if (result)
          Abc_Print( 1, "The command finished successfully.\n" );
        else Abc_Print( 1, "The command execution has failed.\n" );

    }
    return 0;
  usage:
    Abc_Print(-2, "usage: firstcmd [-vh]\n" );
    Abc_Print(-2, "\t First command in ABC\n" );
    Abc_Print(-2, "\t-v : toggle printing verbose information [default = %s]\n", fVerbose ? "yes" : "no" );
    Abc_Print(-2, "\t-h : print the command usage\n" );
    return 1;
}
```

File: src/base/main/mainInit.c

- Include the new command



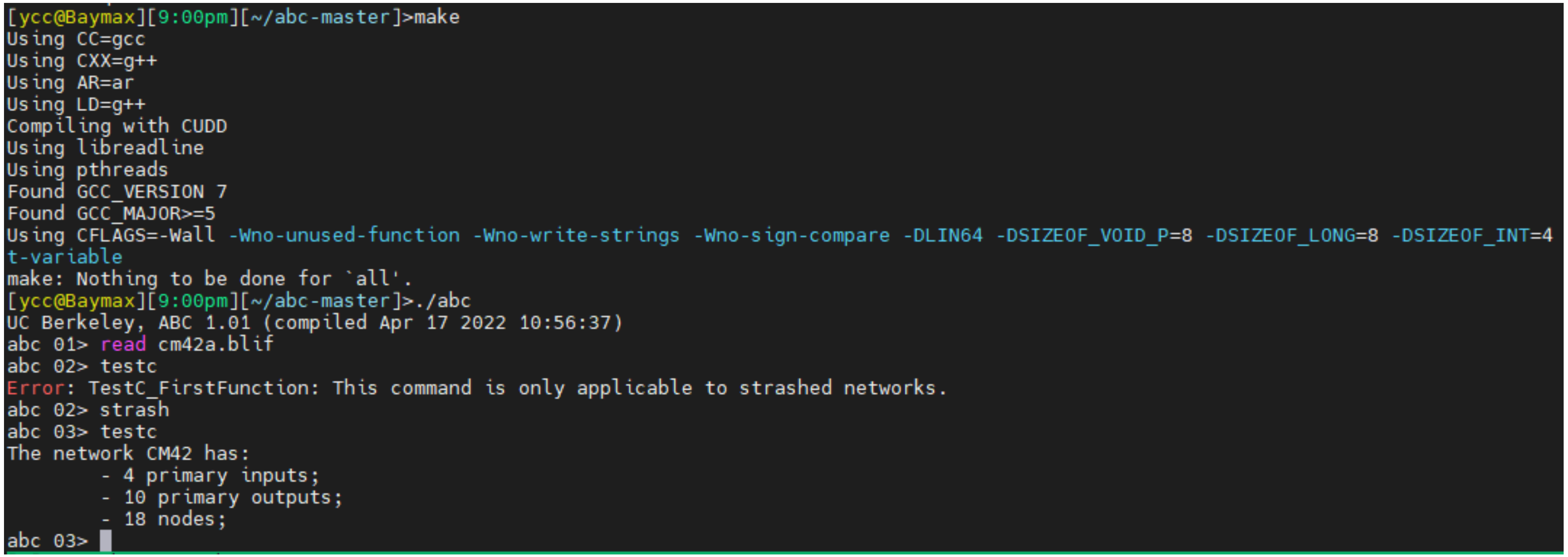
```c
extern void Glucose2_Init( Abc_Frame_t *pAbc );
extern void Glucose2_End( Abc_Frame_t * pAbc );

extern void TestC_Init(Abc_Frame_t * pAbc);

static Abc_FrameInitializer_t* s_InitializerStart = NULL;
static Abc_FrameInitializer_t* s_InitializerEnd = NULL;
```



```c
    Test_Init( pAbc );
    Glucose_Init( pAbc );
    Glucose2_Init( pAbc );
    TestC_Init(pAbc);
    for( p = s_InitializerStart ; p ; p = p->next )
        if(p->init)
            p->init(pAbc);
}
```

Test New Command

- Type make to recompile ABC
- Start ABC, read a network and test the command



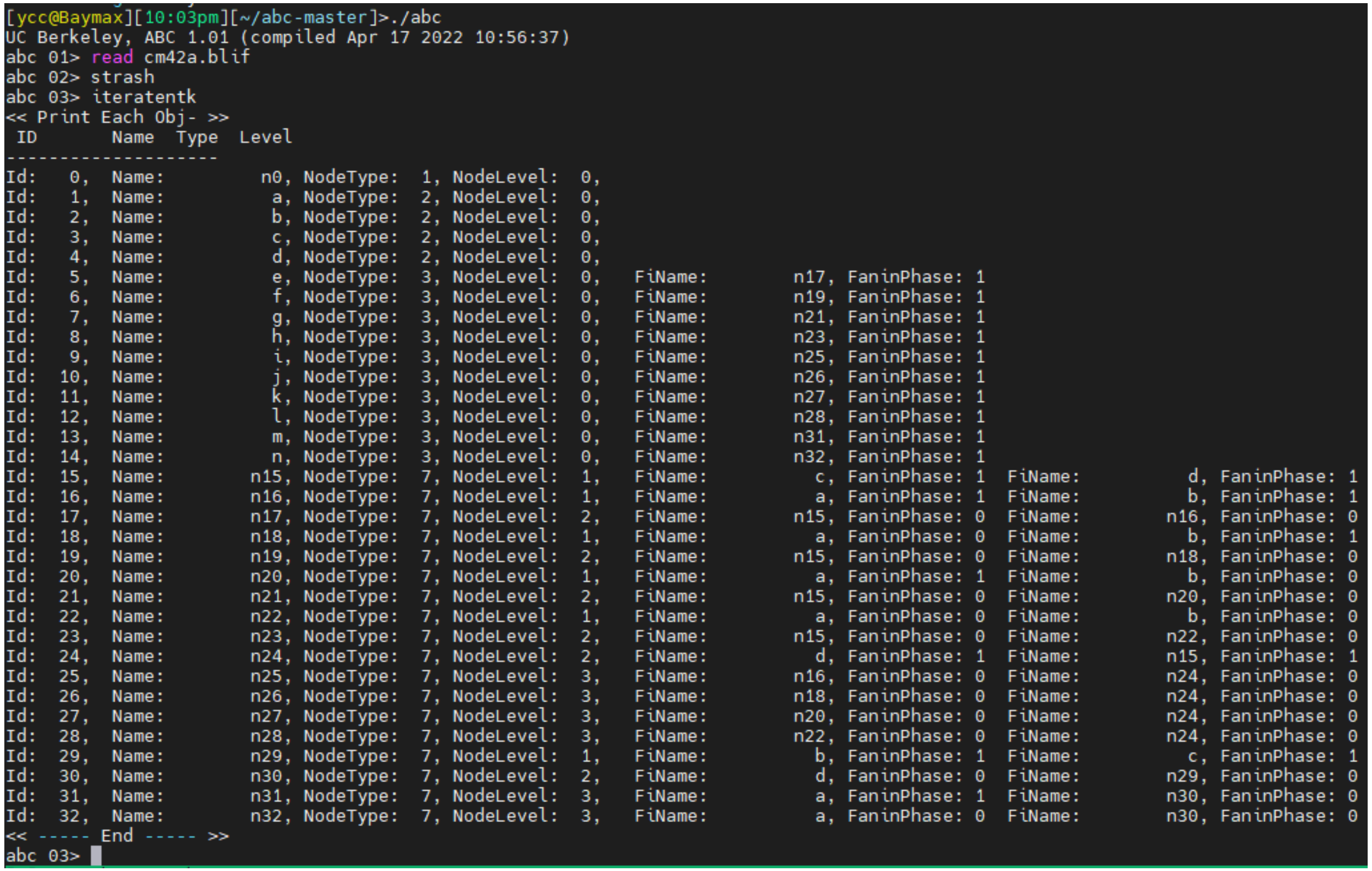
```
[ycc@Baymax][9:00pm][~/abc-master]>make
Using CC=gcc
Using CXX=g++
Using AR=ar
Using LD=g++
Compiling with CUDD
Using libreadline
Using pthreads
Found GCC_VERSION 7
Found GCC_MAJOR>=5
Using CFLAGS=-Wall -Wno-unused-function -Wno-write-strings -Wno-sign-compare -DLIN64 -DSIZEOF_VOID_P=8 -DSIZEOF_LONG=8 -DSIZEOF_INT=4
t-variable
make: Nothing to be done for `all'.
[ycc@Baymax][9:00pm][~/abc-master]>./abc
UC Berkeley, ABC 1.01 (compiled Apr 17 2022 10:56:37)
abc 01> read cm42a.blif
abc 02> testc
Error: TestC_FirstFunction: This command is only applicable to strashed networks.
abc 02> strash
abc 03> testc
The network CM42 has:
        - 4 primary inputs;
        - 10 primary outputs;
        - 18 nodes;
abc 03>
```

Programming Assignment 1

- Write a procedure in ABC environment to iterate over the objects of the AIG network
- Integrate this procedure into an ABC new command “iteratentk”, so that running command " iteratentk " would invoke your code, and print the result

- More benchmarks
  - https://ddd.fit.cvut.cz/www/prj/Benchmarks/

Example



```
[ycc@Baymax][10:03pm][~/abc-master]>./abc
UC Berkeley, ABC 1.01 (compiled Apr 17 2022 10:56:37)
abc 01> read cm42a.blif
abc 02> strash
abc 03> iteratentk
<< Print Each Obj- >>
 ID        Name  Type  Level
--------------------
Id:   0,  Name:        n0, NodeType:  1, NodeLevel:  0,
Id:   1,  Name:         a, NodeType:  2, NodeLevel:  0,
Id:   2,  Name:         b, NodeType:  2, NodeLevel:  0,
Id:   3,  Name:         c, NodeType:  2, NodeLevel:  0,
Id:   4,  Name:         d, NodeType:  2, NodeLevel:  0,
Id:   5,  Name:         e, NodeType:  3, NodeLevel:  0,   FiName:        n17, FaninPhase: 1
Id:   6,  Name:         f, NodeType:  3, NodeLevel:  0,   FiName:        n19, FaninPhase: 1
Id:   7,  Name:         g, NodeType:  3, NodeLevel:  0,   FiName:        n21, FaninPhase: 1
Id:   8,  Name:         h, NodeType:  3, NodeLevel:  0,   FiName:        n23, FaninPhase: 1
Id:   9,  Name:         i, NodeType:  3, NodeLevel:  0,   FiName:        n25, FaninPhase: 1
Id:  10,  Name:         j, NodeType:  3, NodeLevel:  0,   FiName:        n26, FaninPhase: 1
Id:  11,  Name:         k, NodeType:  3, NodeLevel:  0,   FiName:        n27, FaninPhase: 1
Id:  12,  Name:         l, NodeType:  3, NodeLevel:  0,   FiName:        n28, FaninPhase: 1
Id:  13,  Name:         m, NodeType:  3, NodeLevel:  0,   FiName:        n31, FaninPhase: 1
Id:  14,  Name:         n, NodeType:  3, NodeLevel:  0,   FiName:        n32, FaninPhase: 1
Id:  15,  Name:       n15, NodeType:  7, NodeLevel:  1,   FiName:          c, FaninPhase: 1  FiName:          d, FaninPhase: 1
Id:  16,  Name:       n16, NodeType:  7, NodeLevel:  1,   FiName:          a, FaninPhase: 1  FiName:          b, FaninPhase: 1
Id:  17,  Name:       n17, NodeType:  7, NodeLevel:  2,   FiName:        n15, FaninPhase: 0  FiName:        n16, FaninPhase: 0
Id:  18,  Name:       n18, NodeType:  7, NodeLevel:  1,   FiName:          a, FaninPhase: 0  FiName:          b, FaninPhase: 1
Id:  19,  Name:       n19, NodeType:  7, NodeLevel:  2,   FiName:        n15, FaninPhase: 0  FiName:        n18, FaninPhase: 0
Id:  20,  Name:       n20, NodeType:  7, NodeLevel:  1,   FiName:          a, FaninPhase: 1  FiName:          b, FaninPhase: 0
Id:  21,  Name:       n21, NodeType:  7, NodeLevel:  2,   FiName:        n15, FaninPhase: 0  FiName:        n20, FaninPhase: 0
Id:  22,  Name:       n22, NodeType:  7, NodeLevel:  1,   FiName:          a, FaninPhase: 0  FiName:          b, FaninPhase: 0
Id:  23,  Name:       n23, NodeType:  7, NodeLevel:  2,   FiName:        n15, FaninPhase: 0  FiName:        n22, FaninPhase: 0
Id:  24,  Name:       n24, NodeType:  7, NodeLevel:  2,   FiName:          d, FaninPhase: 1  FiName:        n15, FaninPhase: 1
Id:  25,  Name:       n25, NodeType:  7, NodeLevel:  3,   FiName:        n16, FaninPhase: 0  FiName:        n24, FaninPhase: 0
Id:  26,  Name:       n26, NodeType:  7, NodeLevel:  3,   FiName:        n18, FaninPhase: 0  FiName:        n24, FaninPhase: 0
Id:  27,  Name:       n27, NodeType:  7, NodeLevel:  3,   FiName:        n20, FaninPhase: 0  FiName:        n24, FaninPhase: 0
Id:  28,  Name:       n28, NodeType:  7, NodeLevel:  3,   FiName:        n22, FaninPhase: 0  FiName:        n24, FaninPhase: 0
Id:  29,  Name:       n29, NodeType:  7, NodeLevel:  1,   FiName:          b, FaninPhase: 1  FiName:          c, FaninPhase: 1
Id:  30,  Name:       n30, NodeType:  7, NodeLevel:  2,   FiName:          d, FaninPhase: 0  FiName:        n29, FaninPhase: 0
Id:  31,  Name:       n31, NodeType:  7, NodeLevel:  3,   FiName:          a, FaninPhase: 1  FiName:        n30, FaninPhase: 0
Id:  32,  Name:       n32, NodeType:  7, NodeLevel:  3,   FiName:          a, FaninPhase: 0  FiName:        n30, FaninPhase: 0
<< ----- End ----- >>
abc 03>
```

Programming Help

♦ Example of code to iterate over the objects

```
void Abc NtkCleanCopy( Abc Ntk t * pNtk ) {
        Abc Obj t * pObj;
        int i;
        Abc NtkForEachObj( pNtk, pObj, i )
                pObj->pCopy = NULL;
}
```

Refer to src/base/abc/abc.h to find the functions you need

Delivery & Due Date

- The new folder you create and all the files in it.
- Screenshot of ABC running your new command "iteratentk" as shown on the previous page

- Due on 2024/3/14 before class starts